Determining the Effectiveness of the Usability Problem Inspector: A Theory-Based Model and Tool for Finding Usability Problems

Terence S. Andre

Major, USAF

2000

275 pages

Doctor of Philosophy
in
Industrial and Systems Engineering

Virginia Polytechnic Institute and State University

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
15.Jun.00

3. REPORT TYPE AND DATES COVERED
DISSERTATION

4. TITLE AND SUBTITLE
DETERMINING THE EFFECTIVENESS OF THE USABILITY PROBLEM INSPECTOR: A THEORY-BASED MODEL AND TOOL FOR FINDING USABILITY PROBLEMS

5. FUNDING NUMBERS

6. AUTHOR(S)
MAJ ANDRE TERENCE S

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
VIRGINIA POLYTECHNICAL INSTITUTE

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
THE DEPARTMENT OF THE AIR FORCE
AFIT/CIA, BLDG 125
2950 P STREET
WPAFB OH 45433

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
FY00-206

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT
Unlimited distribution
In Accordance With AFI 35-205/AFIT Sup 1

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

14. SUBJECT TERMS

15. NUMBER OF PAGES
275

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT

18. SECURITY CLASSIFICATION OF THIS PAGE

19. SECURITY CLASSIFICATION OF ABSTRACT

20. LIMITATION OF ABSTRACT

Determining the Effectiveness of the Usability Problem Inspector: A Theory-Based Model and Tool for Finding Usability Problems

Terence S. Andre

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Industrial and Systems Engineering

APPROVED:

H. Rex Hartson

Robert C. Williges, Co-Chair

H. Rex Hartson, Co-Chair

Thomas A. Dingus

Thomas A. Dingus

Brian M. Kleiner

Justin D. Rueb

Justin D. Rueb

April, 2000
Blacksburg, Virginia

Keywords: Usability Evaluation Methods, Usability Inspection, Evaluation Effectiveness, Human-Computer Interaction

Determining the Effectiveness of the Usability Problem Inspector: A Theory-Based Model and Tool for Finding Usability Problems

Terence S. Andre

(ABSTRACT)

The need for cost-effective usability evaluation has led to the development of methodologies to support the usability practitioner in finding usability problems during formative evaluation. Even though various methods exist for performing usability evaluation, practitioners seldom have the information needed to decide which method is appropriate for their specific purpose. In addition, most methods do not have an integrated relationship with a theoretical foundation for applying the method in a reliable and efficient manner. Practitioners often have to apply their own judgment and techniques, leading to inconsistencies in how the method is applied in the field. Usability practitioners need validated information to determine if a given usability evaluation method is effective and why it should be used instead of some other method. Such a desire motivates the need for formal, empirical comparison studies to evaluate and compare usability evaluation methods. In reality, the current data for comparing usability evaluation methods suffers from a lack of consistent measures, standards, and criteria for identifying effective methods.

The work described here addresses three important research activities. First, the User Action Framework was developed to help organize usability concepts and issues into a knowledge base that supports usability methods and tools. From the User Action Framework, a mapping was made to the Usability Problem Inspector; a tool to help practitioners conduct a highly focused inspection of an interface design. Second, the reliability of the User Action Framework was evaluated to determine if usability practitioners could use the framework in a consistent manner when classifying a set of usability problems. Third, a comprehensive comparison study was conducted to determine if the Usability Problem Inspector, based on the User Action Framework, could produce results just as effective as two other inspection methods (i.e., the heuristic evaluation and the cognitive walkthrough). The comparison study used a new comparison approach with standards, measures, and criteria to prove the effectiveness of methods. Results from the User Action Framework reliability study showed higher agreement scores at all classification levels than was found in previous work with a similar classification tool. In addition, agreement using the User Action Framework was stronger than the results obtained from the same experts using the heuristic evaluation. From the inspection method comparison study, results showed the Usability Problem Inspector to be more effective than the heuristic evaluation and consistent with effectiveness scores from the cognitive walkthrough.

ACKNOWLEDGMENTS

I would like to take this opportunity to gratefully acknowledge many individuals that helped make my graduate experience very rewarding. First, I am indebted to my wife (Debbie) and my two children (Jennifer and Brian). Graduate school can be a very lonely endeavor, but I am so thankful I was able to share the experience with my family. They cheered me on and provided a place where I could escape the daily grind of research and course assignments. I was not always able to clear my mind of the academic issues, but they accepted my distracted attention and allowed me to continue working when they really deserved my full attention.

I also benefited from the support and professionalism of my co-chairs, Dr. Robert C. Williges and Dr. H. Rex Hartson. From the first day of graduate school, Dr. Williges provided me with the needed environment for conducting effective independent research. He also recognized the importance of the Air Force sponsorship driving my graduate experience and kept me focused on meeting all of their requirements in a timely manner. Dr. Williges also took the time to include my family at every point along the way with interest in our activities and my career with the Air Force. As the co-chair from the Computer Science Department, Dr. Hartson provided the initial idea and motivation for the research effort. I am indebted to him for all the times he rolled up his sleeves and worked alongside me to produce the best product possible. I am grateful for the times he allowed me to be a part of his own writing and idea development processes. His personal mentoring will forever impact the way I go about tackling difficult research problems.

In addition to two great co-chairs, I was also blessed with a great committee. Dr. Dingus and Dr. Kleiner provided very important perspectives from their own work that shaped the construction and scope of the dissertation work. Dr. Rueb from the Air Force Academy provided a great balance to the research effort and ensured the dissertation writing and statistical conclusions were absolutely correct at every point. I am grateful for the enormous time he spent reviewing every single page of the dissertation and providing constructive revisions.

Lastly, I would like to thank two fellow graduate students, Steve Belz and Faith McCreary, for going the research road together. Their personal interest in the project helped establish a significant contribution for determining the reliability of the framework. I am thankful for their friendship and team contributions that made the work an enjoyable journey.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1. INTRODUCTION

PROBLEM STATEMENT

Although software developers recognize the importance of usability evaluation, the cost is relatively high and developers are often unsure about the effectiveness or usefulness of the results from evaluation methods. Most usability evaluation methods (UEMs) are not highly focused, provide little to no theoretical framework for applying the method in a reliable manner, and do not necessarily identify problems worth solving. In addition, even though researchers have conducted numerous studies to compare UEMs, these studies do not provide adequate standards, measures, or criteria to compare one method with another. As a result, the body of literature regarding comparisons of usability evaluation methods is an assortment of conclusions that do not always have clear empirical support.

GOALS

The work described in this research focused on three goals in the development of new methods for evaluating user interaction design. First, develop a usability inspection method that is both cost effective and has utility for finding problems that are worth solving. Second, develop a UEM comparison approach with standards, measures, and criteria to prove the effectiveness of methods. Third, apply the UEM comparison approach by evaluating the effectiveness of a new expert-based inspection method. The work accomplished through these research goals is expected to lead to a more effective and reliable inspection method for usability practitioners. In addition, this work is expected to provide researchers with a method for conducting UEM comparison studies using standardized measures and definitions.

APPROACH

Williges, Williges, and Elkerton (1987) noted the importance of a systematic process for conducting human-computer interaction (HCI) research. The research described in this work mirrors the design stages outlined by Williges and Hartson (1986) and Williges et al. The stages include: (1) *initial design*, focusing on objectives of the research; (2) *formative evaluation* to examine if early research concepts are moving closer to accomplishing the goals; and (3) *summative evaluation* using experimental procedures to compare to other methods. These design stages provide the process for accomplishing the two major goals: developing a cost-effectiveness usability inspection method and developing a UEM comparison approach.

Developing a Cost-Effective Usability Inspection Method

Highly Focused Evaluation

The current demand to get products to market quickly presents developers with an important challenge. Time-intensive methods such as lab-based usability testing and the cognitive walkthrough are not practical for most developers. In addition, companies developing World Wide Web applications and information sites have even a shorter time to market, sometimes as little as 24 hours. Developers in such a situation have only a couple hours to evaluate (if at all) their design before it is placed on the Web for public use. In many cases, they can only do an individual walkthrough, since access to other people may be limited. For such a developer, reliable methods and tools are needed to help a single person design and inspect within a few hours. Thus, practitioners need an inspection method easily tailored for a range of applications, resulting in a highly focused evaluation of the critical aspects of the user interaction design.

Integrating Framework and Theory

Practitioners of HCI are moving away from an emphasis on generalized support methods to systematic evaluation methods in order to help provide reliable evaluation results. The cognitive walkthrough is the only inspection method based on theory to help support a systematic evaluation. The cognitive walkthrough is based on a theory of exploratory learning geared toward walk-up-and-use systems (Polson & Lewis, 1990). However, the evidence suggests that the cognitive walkthrough has not proved as effective as hoped. May and Barnard (1995) argue the problem lies not with the cognitive walkthrough or its underlying theory in particular, but with its limited scope and in the increasing dissociation of an evaluation method from its theoretical foundation – a common problem for methods that have a theoretical foundation. What is needed is an inspection method that has an integrating framework and an easy-to-use theoretical foundation.

Since most methods do not provide a framework or systematic method to guide and structure the capture and reporting of usability problem information, much of the originally available information is lost. Providing a framework of usability attributes minimizes individual differences in reporting through a standardized process for developing usability problem descriptions, which (1) are complete in terms of the attributes applicable to a problem type, and (2) distinguish a problem of one type from a problem of another type. Another important reason for an integrating framework is for the purpose of comparing evaluation results across studies. Many studies have compared different HCI evaluation techniques (heuristic evaluation, usability testing, guidelines, cognitive walkthrough methods, etc.), but have not put their results in a formal framework of HCI. Cuomo and Bowen (1994) have used Norman's (1986) theory of action model in the context of support for a direct manipulation framework, but the integration and classification of data came after the evaluation; it was not built into the method used by evaluators. Thus, much of the original information found by the evaluators was lost in the process.

Theoretical foundations that are easy to understand and use by practitioners have a greater chance of providing effective results. What is currently needed is an evaluation method retaining a theoretical element to provide the necessary conceptual support during evaluation. The appropriate conceptual support will enable designers to identify, comprehend, and resolve usability problems, and would also be less limited than dissociated evaluation methods in their breadth and depth of application.

Effectiveness of Evaluation Methods

The goal of both formal lab-based usability testing with users and expert-based inspection is the same: to improve the usability of products (Rosenbaum, 1989). However, formal lab-based usability testing is not always a feasible alternative. Facilities for testing may not be available. The product may not be mature enough to test, or a software designer may need feedback sooner than is possible with current usability testing. Expert-based inspection methods, in some cases, provide a reasonable alternative, with the heuristic evaluation and the cognitive walkthrough currently considered the leading inspection methods.

A defining characteristic of expert-based inspection methods is they draw on expert knowledge to provide judgments about system usability (Dutt, Johnson, & Johnson, 1994). Expert-based inspection methods can be used effectively when the objectives of the study are to identify usability problems and choose from among the design alternatives in the early stages of the product development cycle (Brooks, 1994). Unfortunately, current expert-based inspection methods fall short of both developer and evaluator needs. First, inspection methods are not as good as user testing for understanding trade-offs (i.e., which dimensions are most important to users). Second, even though inspection methods are excellent at finding potential problems, they are generally ineffective at driving the design of improved user interfaces. Third, except for the heuristic evaluation, most inspection methods require a significant amount of training and analysis time. Finally, inspection methods focus on one particular interface style (e.g., graphical user interfaces) and are difficult to adapt to other possible interfaces (e.g., voice, Web, and virtual environments).

Usability of Methods

Usability of methods is another important concern in the development of effective inspection techniques. Learnability and usability of an inspection technique has been a frequent complaint of many researchers and practitioners (Mack & Nielsen, 1994). Some methods require extensive learning (e.g., cognitive walkthroughs) going beyond the resources many development organizations have in terms of time and personnel. Therefore, a method with an integrating framework and theoretical foundation should also be developed with its own usability in mind.

Developing a UEM Comparison Approach

Usability practitioners are interested in identifying which UEMs are most effective and under which conditions. Developers are generally concerned about time and cost as well as knowing that problems identified by any evaluation method are worth solving and lead to improved design. The need to evaluate and compare UEMs is highlighted by the fact that some researchers have recently questioned the effectiveness of some types of UEMs in terms of their ability to predict problems that users actually encounter (John & Marks, 1997). How can practitioners know if a given UEM is effective and why it should be used over some other UEM? Answering such questions have been difficult because research results from UEM studies have had mixed results in terms of providing conclusive statements about effectiveness. In UEM studies, it is not uncommon to find one researcher showing an expert-based inspection method to be twice as effective as usability testing (Jeffries, Miller, Wharton, & Uyeda, 1991), while another finds usability testing more effective than an expert-based method (Karat, 1994).

Researchers offer various reasons for the mixed results. For example, May and Barnard (1995) argue that ineffective results from the use of an expert-based inspection method is due to the increasing dissociation of the method from its theoretical foundation, if it has one. Although their argument is reasonable, researchers have not conducted studies to support this contention. John and Marks (1997) offer a more supportable hypothesis, suggesting that many methods are not used as intended. For example, the cognitive walkthrough (Polson, Lewis, Rieman, & Wharton, 1992a; Wharton, 1992; Wharton, Rieman, Lewis, & Polson, 1994) represents the most time-consuming expert-based inspection method and is known to provide poor results when used by inexperienced evaluators or when tasks are not clearly defined (Desurvire, 1994).

Examining UEM comparison studies in closer detail highlight the fact that researchers are not able to provide conclusive evidence in support of a particular UEM. Part of the difficulty with results from UEM studies lies in the lack of consistent metrics to measure effectiveness. In addition, very few studies actually identify the target criteria against which to measure success of the UEM being examined. Thus, research concerning the effectiveness of UEMs should include standard metrics and clearly defined criteria so that practitioners can make better decisions when selecting a particular method based on results reported in the literature.

BACKGROUND

What is Usability?

Methodology, theory, and practice in the field of HCI all share the goal of producing interactive software that can be used efficiently, effectively, safely, and with satisfaction. Developing usable products is no longer seen as a "nice-to-have" in the world of product development, but rather as a "must-have." Usability has now grown to be part of every good effort to release a product that consumers can easily learn and use. To most users, the interface is the system; communication with the system has become at least as important as computation by the system (Hartson, 1998). Karat (1997b) points out that the usability of a product is not an attribute of the product alone; it is an attribute of interaction with a product in a context of use. As defined by Nielsen (1993), usability has multiple components and is traditionally associated with five attributes: (1) learnability, (2) efficiency once the system is learned, (3) ability of users to infrequently return to the system without having to relearn, (4) low error rate, and (5) satisfaction. Motivation to deliver these attributes to the consumer has led to the development of large usability testing environments and associated methods to design for these five usability characteristics.

User Interaction vs. User Interface Software

An important distinction to make when considering usability methods is the difference between user interaction and software. Many developers think of their usability methods efforts in terms of "evaluating software" or "evaluating user interface software." Software is the code running behind the interface screen and can be evaluated, but not for the purpose of usability. A more appropriate focus is shown on the left side of Figure 1-1 where usability is seated within the design of the *user interaction component* of an interactive system and not within the *user interface software component* (Hartson, 1998; Hix & Hartson, 1993). The view of the user interaction component is *the user's perspective* of user interaction: how it works, how tasks are performed using it, its look and feel and behavior in response to what a user sees and hears and does while interacting with the computer (Hartson). This perspective is quite different from the user interface software component, which is essentially the programming code by which the interaction component is implemented. According to Hartson, *the user interaction component design can serve as requirements for the user interface software component.* Design of the user interaction component must be given attention at least equal to that given the user interface software component during the development process, if we are to ensure usability in interactive systems.

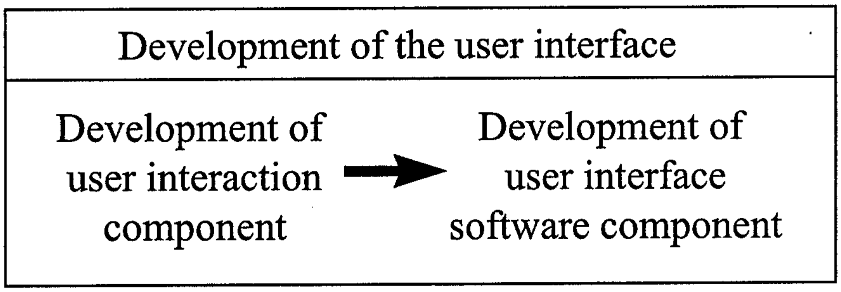

Development of the user interface
Development of user interaction component
Development of user interface software component

FIGURE 1-1. Separating Interaction Development from User Interface Software (Hartson, 1998).

Usability Engineering

Attempts to elevate standard practice grew, when in the late 1980s, several researchers formalized the focus of usability by introducing a formal term and process called *usability engineering* (Whiteside, Bennett, & Holtzblatt, 1988). Wixon and Wilson (1997) define usability engineering as a process for defining, measuring, and thereby improving, the usability of products. Researchers who have contributed to the concept of usability engineering include Bennett (1984), Butler (1996), Gould (1988), and Nielsen (1993). Usability engineering has been most rigorously developed for software design (e.g., Nielsen) and involves four general approaches to design:

- Early focus on the user and tasks,
- Empirical measurement using questionnaires, usability studies, and usage studies focusing on quantitative performance data,
- Iterative design using prototypes, where rapid changes are made to the interface design, and
- Participatory design where users are directly involved as part of the design team.

Based on experience with the usability engineering process at Digital Equipment Corporation, Wixon and Wilson (1997) made three conclusions about usability. First, a traditional experimental approach was inadequate for designing user interfaces. While usability testing is a powerful process, it is cumbersome to track in the fast-paced development process. Second, to be treated seriously, usability efforts had to adopt the assumptions and language of engineering. Third, for usability to be taken seriously, it has to be treated as part of engineering quality, elevating it to the level of other engineering qualities like reliability and performance. The primary rationale for usability engineering is to quantify usability so that it is not just personal opinion. Wixon and Wilson argue that usability "bugs" can be just as severe as reliability bugs and should be treated with the same quantitative respect.

Applying usability engineering has shown to be a worthwhile investment for most development efforts. Karat's (1997a) work in cost-benefit analysis has shown up to a $10 return for every dollar invested in usability. Generally, the value of usability engineering has surfaced in five areas: (1) higher user satisfaction, (2) increased sales, (3) better documentation, (4) reduced training costs, and (5) cost avoidance in support, service, and maintenance. Wixon and

Wilson (1997) document a case study of a usability-engineered product that achieved revenues that were 80% higher than for the first release developed without usability engineering. Interviews showed that buying decisions were made based on usability. In addition, Wixon and Wilson showed that 80% of the software life cycle costs are spent in the post-release maintenance phase.

One of the major challenges in applying usability engineering concerns the effective combination of methods into an overall approach compatible with the constraints of the design problem and the development situation. Wixon and Wilson (1997) recommend treating usability engineering as a high-level process into which other methods can be integrated (object-oriented design, cost benefit analysis, contextual inquiry, inspections, design reviews, scenarios, thinking aloud, metaphor brainstorming, questionnaires, focus groups, user feedback).

Usability Evaluation Methods

Evaluation is an integral part of any development process, whether for considerations of cost, safety, reliability, maintainability, or validation of performance requirements. Definitions frequently vary as to what evaluation really means, and are sometimes discipline specific. In military systems, evaluation infers testing system performance against operational requirements. For many commercial systems, evaluation may refer to conformance with design specifications. Whitefield, Wilson, and Dowell (1991) suggest that evaluation in the human factors discipline involves an assessment of the conformity between a system's performance and its desired performance. In many cases, practitioners may assume that evaluation is a single event that occurs at or near the end of the design process. However, this assumption is not correct. For many years now, the HCI community, as well as other disciplines, has recognized that evaluation occurs throughout the design life cycle, with results feeding back into modifications to the design (Dix, Abowd, & Beale, 1993). Usability evaluation in essence is a continual process of refining the design of the user interaction component based on frequent inputs from stakeholders (e.g., designers, expert evaluators, and users of the system). Researchers in the field of HCI have developed various UEMs as part of the usability engineering process. In the context of this dissertation, UEMs refer to any method or technique used to perform formative usability evaluation (any kind of usability evaluation used to improve usability) of an interaction design at any stage of its development. Thus, lab-based usability testing with users, heuristic evaluation, and other expert-based usability inspection methods are included in the scope of improving usability evaluation. Essentially, this approach includes every method used to produce a list of usability problems as its output.

In the late 1980s and early 1990s, laboratory usability testing quickly became the primary usability evaluation method for examining a new or modified interface. Laboratory usability testing was seen by developers as a way to minimize the cost of potential service calls, increase sales through the design of a more competitive product, minimize risk, and create a historical record of usability benchmarks for future releases (Rubin, 1994). In the last few years, many developers have explored other methods in an attempt to bring down the cost and time requirements of traditional usability testing. In addition, because usability testing often occurs late in the design process, developers were motivated to look at methods that could be used earlier when only an immature design was available (Marchetti, 1994). As a result, expert-based inspection methods grew in popularity because many of them were intended to be used with a relatively early design concept (Bradford, 1994).

OUTLINE OF RESEARCH

In order to meet the goals of this research, a systematic iterative design process was used to first develop the necessary tools and methods followed by a summative study to establish the reliability of the new evaluation framework and compare it to other established methods in the field. Work conducted to meet the research goals include the following activities:

(1) Background Literature

- Outline the strengths and weaknesses of current evaluation methods used to find usability problems; identifying the factors that contribute to effective evaluation and those that seem to contaminate the process. [Chapter 2]
- Discuss candidate measures for determining effectiveness of methods; identifying the factors that contribute to effective evaluation and those that seem to contaminate the process. [Chapter 2]

(2) Development of the User Action Framework and the Usability Problem Inspector

- Develop the User Action Framework based on background literature and successes and failures from current usability methods. Includes an emphasis on linking framework and theory to the design process, developing a method compatible with the theory, and generic structure so that mapping to different styles (e.g., voice, Web, virtual reality) can be accomplished without "deforming" the framework. [Chapter 3]

- Develop the Usability Problem Inspector tool from the general integrating framework. Examine platforms for easy access to practitioners (e.g., web-based tool). Implement an inspection method that reduces training time to an acceptable level and is easy to use and understand. [Chapter 3]

(3) Pilot Study on the Usability Problem Inspection Tool

- Conduct a pilot study on an early version of the Usability Problem Inspector to help provide feedback to the formative evaluation process. [Chapter 4]

(4) Test Reliability of the Framework

- Perform a reliability test of the User Action Framework before the final inspection tool is implemented. The purpose of the reliability test is to document the reliability of the User Action Framework in helping evaluators locate a given usability problem in the underlying framework. [Chapter 5]

(5) Formal Comparison Study

- Conduct a formal, summative study to compare the Usability Problem Inspector with the heuristic evaluation technique and the cognitive walkthrough, using a lab-based usability test for baseline comparison. Develop Venn diagrams on how the methods compare. Compare measures such as thoroughness, validity, effectiveness, and user reports assessing the overall quality of the methods. [Chapter 6]

CHAPTER 2. BACKGROUND LITERATURE

The evaluation of user interaction design is a rather large research area with many different goals, methods, and research implications. This section discusses UEMs used to perform formative usability evaluation of an interaction design at any stage of its development. A complete understanding of the various UEMs is needed in order to identify the gaps in this research area. In addition, measuring the effectiveness of UEMs is addressed to develop an understanding of the limitations of current metrics and what steps are needed to improve our approach to examining effectiveness for current and future UEMs.

USABILITY EVALUATION METHODS

Researchers classify UEMs on many different dimensions. Nielsen and Molich (1990) discuss four basic approaches to usability evaluation. These include automatic, formal, empirical, and heuristic methods. According to Nielsen and Molich, automated approaches have been limited to a few primitive computerized checks of interface elements. Formal methods include cognitive modeling approaches such as GOMS (Card, Moran, & Newell, 1983), Cognitive Complexity Theory (Kieras & Polson, 1985), and the CE+ Model (Polson & Lewis, 1990). Empirical methods focus on observing users as they perform tasks and include usability testing, field testing, and attitude questionnaires. Heuristic methods attempt to reduce the time and cost associated with evaluation and include such methods as heuristic evaluation and guideline reviews. Wixon and Wilson (1997) proposed five dimensions along which usability methods can be classified. The five dimensions include: (1) formative versus summative, (2) discovery versus decision, (3) formalized versus informal, (4) designer involvement versus user involvement, and (5) complete and component. Whitefield, Wilson, and Dowell (1991) provided a framework for human factors evaluation using four categories: (1) analytic methods, (2) specialist reports, (3) user reports, and (4) observational methods.

Differences between authors in classifying evaluation methods rightly reflect the different perspectives one can take when attempting to categorize evaluation methods. Classifying UEMs on a particular dimension (e.g., formative vs. summative, formal, vs. informal) is difficult because practitioners can easily use many of the methods throughout the various stages of the design cycle as well as with different levels of formality. For the research presented here, the purpose is to describe UEMs by dividing them into three traditional categories: (1) empirical methods, (2) expert-based usability inspection methods, and (3) model-based approaches. Table 2-1 shows an overview of the specific methods covered in this research. Although the background presented here could point to specific methods that seem more favorable to practitioners, the real focus is on describing the various methods to better understand the strengths and weaknesses. Better understanding of the strengths and limitations of UEMs can help to develop better tools and techniques that integrate the best of what works in usability evaluation.

TABLE 2-1. Taxonomy of Usability Evaluation Methods.

Category of UEM	**Specific Methods and Techniques**
Empirical Methods	Controlled Experiments Formal Lab-based Usability Testing Field Testing/Operational Evaluation
Expert-Based Usability Inspections	Guideline Reviews Heuristic Evaluation Cognitive Walkthroughs Formal Usability Inspections Usability Walkthroughs Heuristic Walkthroughs
Model-Based Approaches	Stages of User Activity Analysis Model-Mismatch Analysis

Empirical Methods

Controlled Experiments

Conducting controlled experiments in HCI can help us understand the low-level perceptual, cognitive, and motor activities of individuals (Rubin, 1994). Researchers often use formal controlled experiments to conduct basic research where a specific hypothesis is formulated. Experimental designs in HCI involve traditional techniques such as the use of randomly chosen participants, tight controls, control groups, and sample size of users sufficient to measure statistically significant differences between groups (Rubin, 1994). Most experimental design studies in HCI are targeted at determining hard, quantitative data. Frequently, the data are in the form of performance metrics -- how long does it take to select a block of text with a mouse, touchpad, or trackball? How does the placement of the backspace key influence the error rate? Researchers using experimental design techniques are often interested in comparing two systems on a particular characteristic. For example, does the help system as designed in Format A improve the speed and error rate of experienced users more than help as designed in Format B?

Shneiderman (1998) points out that managers of actively used systems are also coming to recognize the power of controlled experiments for fine tuning performance attributes of the user interaction component. Unlike traditional usability evaluation methods, controlled experiments are not used for near-term decisions. Rather, they are used to make long standing decisions about certain features that will be relatively stable over time.

Strengths of Controlled Experiments

Controlled experiments offer the researcher the only method for approaching cause and effect conclusions. By controlling specific variables, the researcher can make solid conclusions about certain design features that are compared in a systematic manner. Controlled experiments also offer more quantitative data than many other methods; providing the necessary justification to convince managers of design decisions.

Weaknesses of Controlled Experiments

As compared to current evaluation approaches, controlled experiments are impossible or inappropriate to use to conduct usability tests in the fast-paced, highly pressurized development environment. The purpose of usability evaluation is not necessarily to formulate and test specific hypotheses, that is, conduct research, but rather to improve products (Rubin, 1994). In addition, finding a large sample size is impractical for most development efforts, especially if several conditions are needed. Controlled experiments are designed to obtain quantitative proof of research hypotheses; that one design is better than another design. They are not designed to obtain qualitative information on how to fix problems and redesign products.

Formal Lab-Based Usability Testing

Lab-based usability testing in computer systems originated during the early development of large computer-based systems at Bell Laboratories in the early 1970s (Bailey, 1972). Since its inception, this type of testing has been successful in the development of numerous computer-based systems. These test programs have led to substantial improvements in human performance in computer products, applications, and systems. The rapid growth of usability testing and laboratories since the early 1980s is an indicator of the attention now focused on designing for user needs (Shneiderman, 1998).

Formal lab-based usability testing is a process that employs participants who are representative of the target population to evaluate the degree to which a product meets specific usability criteria. Usability testing is essentially a research tool, with its roots in classical experimental methodology (Rubin, 1994). The primary purpose of usability testing is to determine whether the product or the process elicits the necessary level of human performance to meet the requirements established for it. When deficiencies or weaknesses are discovered, an opportunity arises for redesign and for retesting of the altered components. Formal lab-based testing provides the developer with a list of problems that impact the user in some significant way. A benefit to HCI practitioners is that a usability test generally identifies problems that will plague the actual users of the application. Researchers and developers need not do extensive filtering of problems according to their predicted impact on users; the impact can usually be assessed from the test (Jeffries & Desurvire, 1992).

Most usability laboratories are equipped to perform controlled hypothesis testing for a single design or competitive testing between two similar products. However, most practitioners today conduct usability testing for the purpose of generating a problem report used to rapidly refine the interaction design. Therefore, unlike controlled experiments, usability testing generally requires fewer subjects (maybe as few as three) and the amount of quantitative performance data are often reduced so the focus can be on errors users find while performing tasks.

As in controlled experiments, usability testing efforts generally follow a well-defined plan for accomplishing the test. Rubin (1994) lists the following six activities that are common to most formal lab-based testing efforts:

- Development of test objectives,
- Use of a representative sample of end users,

- Representation of the actual work environment,
- Observation of end users who either use or review a representation of the product,
- Collection of quantitative and qualitative performance and preference measures, and
- Recommendation of improvements to the design of the product.

Strengths of Usability Testing

Formal lab-based usability testing offers at least two major strengths. First, usability testing is almost an infallible indicator of potential problems and the means to resolve them, when used with the right care and precision, at the appropriate time (Rubin, 1994). Developers have a hard time doubting evidence they can see from a usability test as they review video segments from specific sessions. Second, the process of usability testing is generally well understood and accepted by management as a primary means of minimizing risk of releasing an unstable or unlearnable product.

Weaknesses of Usability Testing

Criticisms of formal lab-based usability testing techniques generally fall into four areas: cost of the test, artificiality of the testing environment, selection of participants, and reliability of the test results. Usability testing is usually considered one of the more expensive evaluation techniques because it requires both user and staff resources (Scerbo, 1995). As with controlled experiments, testing in a usability laboratory does not necessarilly reflect the actual work context and also does not prove that a product works (Rubin, 1994; Whiteside et al., 1988). Some researchers also have doubts as to how well selected participants represent actual users in real work contexts. Holleran (1991) points out that designers can overestimate the power and generalizability of usability tests based on a very small sample of subjects. For these and other reasons, many developers supplement usability testing with other methods such as expert-based usability inspections.

Field Testing

Field tests attempt to put new interfaces to work in realistic environments for a fixed trial period. Most researchers agree that it is valuable to watch people using a new product, application, or system, and to talk with them. Observing users in the field is often the best way to determine their usability requirements. Traditional usability testing, while providing a laboratory environment that makes data collection and recording easy, also removes the user and the product from the context of the workplace. Practitioners and researchers can use field testing as part of the formative evaluation of prototype design. Because the goal of formative evaluation is to guide refinement of a design, and not to generate quantitative comparisons or to support statistical inferences, the amount of data collected and the replicability of the data collection methods are less important than the insights gained (Kies, Williges, & Rosson, 1998). In a field study, the researcher sets out to make direct observations using ethnographic methods. In HCI system development, an experimental prototype can be deployed to a work environment where the researcher collects user logs. Field studies are generally intended to be in settings under conditions as natural as possible.

On-site observation methods are probably the most common way of collecting information about an existing system. Observations are usually conducted while actual users are performing real work. Real-time observations enable evaluators to see how users are interacting with the new system, to ask questions, and to identify any potential usability problems. Researchers often use ethnographic methods in the collection of data when observations are conducted in a field setting.

Surveys or questionnaires collect information from actual users concerning their attitudes about a new system. Researchers can compare information from surveys with the attitudes of users of other products as measured by the same instrument (questionnaire). Also, researchers can measure the attitudes after several months, and then again after one year, to see if there are improvements. Some designers like to collect satisfaction information in a "before-and-after" picture (Bailey, 1997). Designers administer a satisfaction questionnaire to users on the old, existing system (before), and then the exact same questionnaire to users after they are experienced on the new system (after).

A more automated approach to field testing includes the use of user logs in work settings. User logs involve the automatic collection of user interaction events stored by the computer. Researchers typically design the software to capture the number of times each error message appears, the help messages most frequently used, the commands used, the menu items used, which primary windows and dialog boxes are accessed, etc. (Bailey, 1997).

Field researchers are constantly developing new techniques to support field testing and other evaluation activities. For example, Hartson et al. (1996) have shown that remote usability evaluation using the network to capture activities at remote work settings is a very practical solution for evaluation efforts where users are distributed.

Strengths of Field Testing

The primary strength of field testing is its potential to collect data in real work contexts. When combined with automated methods, field testing can be totally unobtrusive (transparent) in that the users do not know that their performance is being monitored. Also, designers can collect data on large numbers of users for a long time, which provides a much better picture of user behaviors as they gain experience (Teubner & Vaske, 1988).

Weaknesses of Field Testing

Even though field testing provides data on the real work context, many developers struggle to find ways to rapidly integrate field testing results with current designs. Most designers apply the information gained from field studies in the next system they develop. Therefore, major changes in user interaction designs occur only for downstream versions. In addition, when not using automated methods, the very process of making observations may distort what is being observed. Users may not be as spontaneous or natural when they are being observed, just as when they are observed in a laboratory setting. In field studies of prototype designs, it is easy to misinterpret observations, disrupt normal practice, and to overlook important information. Using proper ethnographic techniques and remote evaluation approaches can reduce many of these potential weaknesses.

Expert-Based Usability Inspections

Evaluations of software usability, if done early in the design process, can significantly improve the quality of the software. However, practicing software developers seldom employ usability evaluation procedures, because the cost of using them is perceived to be high and the benefits low (Polson, Rieman, Wharton, & Olson, 1992b). Expert-based usability inspection methods are a class of usability evaluation procedures designed to address this criticism.

Usability inspection is relatively new when compared to empirical methods such as lab-based usability testing. Nielsen (1994b) defined expert-based inspection evaluation techniques as a set of methods that rely on evaluators' inspecting usability-related aspects of a user interface. The overall goal of inspection methods is to provide more usable systems at acceptable development costs. For most expert-based usability inspection methods, the focus is on aspects of a design that determine ease of learning and ease of use (Polson et al., 1992b).

In describing usability inspection methods, Virzi (1997) explained that all of the techniques can be considered as a unique combination of three dimensions: (1) the characteristics of the judges, i.e., usability expertise versus application domain knowledge; (2) number of evaluators in a single session, i.e., groups versus individuals; and (3) the goals of the inspection, i.e., finding general usability problems, discovering ease of learning problems, or assessing compliance with a set of standards. Other researchers have suggested additional factors that differentiate the usability inspection methods. For example, Olson and Moran (1996) propose a two-dimensional space that classifies usability inspection methods based on whether they employ guidelines and whether they use scenarios to guide the evaluation. Rather than propose a particular classification scheme for expert-based usability inspection methods, the purpose here is to describe the attributes of each method to help form an understanding of how methods differ.

Guideline Reviews

Guideline reviews are inspections where an interface is checked for conformance with a comprehensive list of usability guidelines (Mack & Nielsen, 1994). Guidelines are typically published in books, reports, and articles that are publicly available. They generally are not specific to a single organization, but rather apply across the broad spectrum of user interaction design (Hix & Hartson, 1993). Therefore, usability practitioners often have to tailor guidelines for their particular organization and product focus.

A classic work in guidelines is the Smith and Mosier (1986) report developed for user interface software. Although some guideline reviews can be accomplished with a short list of heuristic principles, many guideline documents contain on the order of 1,000 guidelines within a document several hundred pages long. Most guideline review efforts are accomplished using a tailored set of guidelines to fit the particular application domain. Guideline reviews are typically conducted with several people from the design team, although they can easily be accomplished by one person on smaller development efforts.

Strengths of Guidelines

The primary strength of guideline reviews is they give the evaluator a methodological approach to inspect the user interface according to the guideline document used by the designer. An additional strength is they allow the evaluator to easily inspeci the design without formal training with the method.

Weaknesses of Guidelines

Most researchers agree that designers and evaluators should not rely solely on guidelines. Guidelines are usually vague, are sometimes contradictory, involving trade-offs between conflicting goals, and are often not translated into specific design rules that can easily be used (Baecker, Grudin, Buxton, & Greenberg, 1995; Blatt & Knutson, 1994). In practical application, designers and evaluators generally find huge sets of guidelines difficult to use. Lund (1997) points out that too many guidelines for interface design exist and that many designers have developed their own lists of general maxims that they keep in working memory. In addition, design guidelines quickly become out dated and have no empirical or theoretical support (Polson et al., 1992a).

Heuristic Evaluation

Heuristic evaluation is probably the most widely known method in the HCI community. Nielsen and Molich (1990) popularized the heuristic technique in the early 90's by developing it as a cheap, fast, and easy-to-use method for inspection of user interfaces. Nielsen (1992) defined heuristic evaluation as a method for finding usability problems in a user interface design by having a small set of evaluators examine the interface and judge its compliance with recognized usability principles (the "heuristics"). Each individual evaluator performs a heuristic evaluation by inspecting the interface alone, keeping in mind a set of usability heuristics (Nielsen, 1994a). These heuristics represent general rules that seem to describe common properties of usable interfaces. Molich and Nielsen (1990) developed the original list of usability heuristics, which Nielsen (1994a) later revised through a factor analysis. The resulting list of usability heuristics is shown in Table 2-2.

TABLE 2-2. Revised Set of Usability Heuristics (from Nielsen, 1994a).

Heuristic
Visibility of system status
Match between system and real world
User control and freedom
Consistency and standards
Error prevention
Recognition rather than recall
Flexibility and efficiency of use
Aesthetic and minimalist design
Help users recognize, diagnose, and recover from errors
Help and documentation

Nielsen (1994b) recommended that the evaluator go through an interface using a two-pass approach. The first pass helps the evaluator get a feel for the system, while the second pass allows the evaluator to focus on specific interface elements. Nielsen also recommended that aggregation of evaluator findings only be done after all individual evaluations have been completed. This approach helps to ensure independent and unbiased evaluations from each evaluator. The result from a heuristic evaluation is a list of usability problems in the interface with reference to the heuristics that were violated in each case.

In conducting usability studies using the heuristic evaluation technique, Nielsen found that usability problems relating to missing interface elements that ought to be introduced were the most difficult to find in interfaces implemented as paper prototypes (Nielsen, 1992). Nielsen (1994b) also concluded that the finding of usability problems and the more detailed analysis of such problems (e.g., severity) are two different processes that should not be interleaved in a single session. Evaluators generally like to focus on finding the problems and not stopping to give severity ratings. Finally, Nielsen (1992) noted that heuristic evaluation is a good method for finding both major and minor problems, but will tend to be dominated by minor problems.

Strengths of Heuristic Evaluation

Heuristic evaluation is meant to be used as a discount usability engineering technique to find usability problems in a product design (Nielsen & Molich, 1990). The major strengths of heuristic evaluation are: it is relatively cheap, it finds lots of problems, it is intuitive and easy to motivate people to use it, it does not require advance planning, and can be used early in the development process when only an immature prototype is available (Nielsen & Molich, 1990). In addition, heuristic evaluation does not require the intensive training and background required by many of the other methods discussed later; although background in cognitive psychology, human-computer interaction, and human factors significantly improves the performance of the evaluator.

Weaknesses of Heuristic Evaluation

Heuristics work well within a limited set of constraints. For example, evaluators should be experts to use the method more effectively, multiple expert evaluations are needed to find a majority of the problems in the interface (i.e., one evaluator will not find all the problems), and evaluators should be given enough time to explore most of the features of the interface (Jeffries & Desurvire, 1992). When these assumptions are not met, heuristic evaluation can become a very ineffective technique. Conclusions from Doubleday, Ryan, Springett, and Sutcliffe (1997) as well as Jeffries, Miller, Wharton, and Uyeda (1991) indicate five major problems with heuristic evaluation: (1) problems can be subjective (on the part of the evaluators using their experience), (2) problems are often not distinct, (3) heuristic evaluators are not as absorbed in using the system as users in the user testing method, (4) it is not task-based, and (5) it identifies a large number of specific, one-time, and low-priority problems. In addition, because heuristics represent general usability principles, evaluators can be easily led to identify false alarms (Sears, 1997). For example, a heuristic such as "provide help" can lead some evaluators to interpret the lack of an explicit help button to be a problem, regardless of the simplicity or purpose of the screen.

Refinements to the Heuristic Evaluation

Several researchers have come along and developed modifications to help improve the heuristic evaluation. Nielsen (1994a) applied a factor analysis to different list of heuristics in order to find the heuristics most related to severe usability problems. Comparing the heuristics explaining the major problems with those explaining the minor problems, Nielsen (1994a) concluded that the heuristics in the top-10 for the major problems that are not in the top-10 for the minor problems are:

- Make the repertoire of available actions salient,
- Prevent errors,
- Easy to discriminate available action alternatives, and
- Modeless interaction.

Nielsen (1994a) recommends closer attention to these heuristics may help increasing the proportion of serious usability problems found by heuristic evaluation

Kurosu, Matsuura, and Sugizaki (1997) proposed a new type of heuristic evaluation method, called the structured heuristic evaluation method (or sHEM). This method is expected to detect more of the usability problems as compared to the conventional heuristic evaluation method introduced by Nielsen and Molich (1990). The sHEM approach stemmed from the idea that there is a trade-off between the number of the activated usability heuristics in the evaluator's memory and the number of problems to be detected by the evaluator. Kuroso et al. proposes solving this trade-off by introducing structure into the set of usability heuristics, e.g., by splitting the evaluation session into several sessions, each of which focuses on only the specific aspect of the whole usability heuristics. Kuroso et al.'s proposed sub-categories, classified by usability concepts and user characteristics, are shown in Table 2-3. Kurosu et al. (1997) have only done a preliminary investigation of these categories; no empirical findings have been reported.

TABLE 2-3. Categories for the Structured Heuristic Evaluation Method (Kurosu et al., 1997).

Usability Concepts	**User Characteristics**
Ease of cognition	Novice vs. experts
Ease of operation	Users with special care
Pleasantness	

Cognitive Walkthrough

The cognitive walkthrough (Lewis, Polson, Wharton, & Rieman, 1990; Polson et al., 1992a) is a usability inspection method that focuses on evaluating a design for ease of learning, particularly by exploration. The focus on ease of learning is motivated by the observation that many users prefer to learn software by exploration (Wharton, Bradford, Jeffries, & Franzke, 1992). In contrast to other types of usability evaluations, the cognitive walkthrough focuses on a user's cognitive activities; specifically, the goals and knowledge of a user while performing a specific task (Wharton et al., 1992). Experts using the cognitive walkthrough evaluate each step necessary to perform a task, attempting to uncover design errors that would interfere with learning by exploration. The method finds mismatches between users' and designers' conceptualization of a task, poor choices of wording for menu titles and button labels, and inadequate feedback about the consequences of an action (Wharton et al., 1994). Because the cognitive walkthrough is based on a cognitive theory of exploration, it requires more knowledge of cognitive science terms, concepts, and skills than most other usability evaluation methods (Lewis et al., 1990; Wharton et al., 1992).

Wharton et al. (1994) described five major steps typically involved in performing the cognitive walkthrough:

1. Define inputs to the walkthrough.
2. Convene the analysts.
3. Walk through the action sequences for each task.
4. Record critical information.
5. Revise the interface to fix the problems.

In the cognitive walkthrough, experts evaluate a proposed interface in the context of specified user tasks. The inputs to a walkthrough session include the identification of users, selection of representative tasks, description of action sequences for completing the tasks, and description or implementation of the interface. Although originally conceived as an independent, individual process, more recent implementations of the cognitive walkthrough include groups of experts that pool their problems into a single report (Lewis, 1997). During the walkthrough process, experts consider each of the user actions needed to accomplish the task. For each action, the experts consider four primary questions intended to stimulate a story about a typical user's interaction with the interface. In particular, the experts ask the following four questions:

1. Will the user try to achieve the right effect?
2. Will the user notice that the correct action is available?
3. Will the user associate the correct action with the effect trying to be achieved?
4. If the correct action is performed, will the user see that progress is being made toward solution of the task? (Wharton et al., 1992).

While performing the evaluation, experts record problems, reasons, and assumptions for each action sequence (Wharton, 1992). In addition, experts construct a success story that explains why a user would choose a particular action, or a failure story to indicate why a user would not choose the action (Lewis, 1997; Wharton et al., 1994). Based on the documentation provided by expert evaluators in the cognitive walkthrough session, designers prioritize problems and modify the interface design to eliminate these problems.

The cognitive walkthrough has several key features that differentiate it from the other expert-based usability inspection methods described in this section. First, the method is based on a theory of learning by exploration (Polson et al., 1992a). Second, the cognitive walkthrough examines specific user tasks, rather than assessing attributes of an interface as a whole (as is done in the heuristic evaluation). Third, the cognitive walkthrough analyzes correct sequences of actions, asking if these correct sequences will actually be followed by users (Polson et al., 1992a). The cognitive walkthrough does not attempt to predict what users will do, beyond suggesting whether they will follow a correct path or depart from it (Lewis, 1997). Thus, the cognitive walkthrough only critiques an action sequence that is provided as input instead of predicting an action sequence. Fourth, the cognitive walkthrough aims not only to identify likely problems in an interface, but also to suggest reasons for these problems (Lewis, 1997).

Studies using the cognitive walkthrough have highlighted the importance of understanding cognitive theory. In fact, the first version of the walkthrough failed when tested with untrained analysts, including students in a user-interface design class and designers in industry (Wharton et al., 1994). Wharton et al. noted that an understanding of terms was very important. For example, analysts without a cognitive science background had trouble distinguishing "goals" from "actions."

Strengths of the Cognitive Walkthrough

The primary strength of the cognitive walkthrough is its task-based approach to evaluating user interaction designs. The focus on specific user tasks is intended to help designers assess how the features of their design fit together to support users' work, or fail to do so. In addition, the focus on critiquing correct action sequences rather than predicting user behavior is intended to provide the most useful feedback for the designer (Lewis, 1997). That is, the cognitive walkthrough provides the designer with information as to how reasonable it is for a user to follow a specific action sequence. Another strength is the ability to permit very early evaluation of designs. Because expert evaluators examine cognitive activities of user tasks, the cognitive walkthrough can be used with very primitive conceptual designs; even paper descriptions. The design must only be mature enough to permit the evaluator to work out correct action sequences for one or more specific user tasks and to envision the cues and responses to be provided by the interface along those sequences (Lewis, 1997).

Weaknesses of the Cognitive Walkthrough

Designers and researchers using the cognitive walkthrough have been very forthcoming in their concerns and weaknesses of the method. Some of the noted concerns include the need for a background in cognitive psychology, the tedious nature of the technique, the types of problems identified, and the extensive time necessary to apply the technique (Lewis et al., 1990; Rowley & Rhoades, 1992; Wharton et al., 1992). Researchers have also pointed out concerns with the scope of the cognitive walkthrough. For example, Jeffries et al. (1991) found cognitive walkthroughs identified more specific problems than general problems. The low-level focus on specific action sequences often leave high-level problems unrecognized. For example, an evaluator might examine a six-step sequence and find appropriate labels and system responses at each step, but fail to notice that the entire sequence could be eliminated by a higher-level change in the interface. John and Mashyna (1997) point out that error recovery is not addressed in the cognitive walkthrough because the focus is on correct action sequences. John and Mashyna suggest that some error recovery tasks should be included in the cognitive walkthrough to reduce this deficiency.

Refinements of the Cognitive Walkthrough

Several researchers have come along to improve certain aspects of the cognitive walkthrough in an attempt to emphasize certain strengths while reducing some of the major weaknesses. For example, Rieman et al. (1991) developed an automated cognitive walkthrough to reduce the time evaluators spent filling out forms. Many of the questions in the automated cognitive walkthrough can be answered by a single mouse click to indicate yes, no, or a percentage; others require a brief text entry. The automated system also helps the analyst to maintain a dynamic description of the hypothetical user's current and active goals. Formal studies of the effectiveness of the automated cognitive walkthrough have not been reported as of yet.

Other researchers have focused on improving the original questions in the cognitive walkthrough. For example, Lavery and Cockton (1997) developed seven questions, extending the original four questions from Wharton et al. (1992) and including some of Norman's theory of action (Norman, 1986). The seven questions that Lavery and Cockton developed are:

1. Will the user try to achieve the right subgoal?
2. What knowledge is needed to achieve the right subgoal? Will the user have this knowledge?
3. Will the user notice that the correct action is available?
4. Will the user associate the correct action with the subgoal they are trying to achieve?
5. Will the user perceive the feedback?
6. Will the user understand the feedback?
7. Will the user see that progress is being made towards solution of their task in relation to their main goal and current subgoals?

Even with the modification and extension of questions, Lavery and Cockton (1997) did not address the important area of error recovery. Verbeek and Van Oostendorp (1998) have also come along to refine and develop more questions for the cognitive walkthrough; using a think-aloud approach to stimulate discussion of possible side issues. Verbeek and Van Oostendorp used the actual users to perform the cognitive walkthrough, but have not reported how this procedural difference compares to the original cognitive walkthrough.

In an attempt to speed up the pace of the cognitive walkthrough, Rowley and Rhoades (1992) developed the cognitive jogthrough for developers who had limited time for the evaluation session. The jogthrough procedure was designed for a product development environment where multi-disciplinary teams evaluate proposed interfaces, but have primary responsibilities other than user interface design and evaluation. Rowley and Rhoades introduced participant roles and a new means for recording the evaluation session. They formally assigned participants to one or two of four roles: evaluator, presenter, moderator, and recorder. Participants met as a group to perform the evaluation. To increase the pace of the evaluation session, the session itself was recorded on videotape. This recording was then used to log significant events, in real time, discussed during the evaluation. Initial feedback from the cognitive jogthrough was positive because more user actions could be covered in the same amount of time normally assigned to the individual walkthrough.

Formal Usability Inspections

Formal usability inspections are intended to be very similar to the code inspection methods used by many software developers. Formal usability inspections were designed to help engineers, who serve as inspectors, to review a product and find a large number of valid usability defects. Most formal usability inspection approaches include aspects of other inspection methods. For example, heuristics help non-usability professionals find usability defects. In addition, inspectors walkthrough tasks with the user's goals and purpose in mind, similar to cognitive walkthroughs, although the emphasis is less on cognitive theory and more on encountering defects. This method formalizes the review of a specification or early prototype. The basic steps are to assemble a team of four to eight inspectors, assign each a special role in the context of the inspection, distribute the design documents to be inspected and instructions, have the inspectors go off on their own to do their inspection, and convene later in a formal inspection meeting (Freedman & Weinberg, 1990).

Kahn and Prail (1994) developed a very thorough formal usability inspection method focusing on users' potential task performance with a product. The Kahn and Prail approach to formal usability inspection included the following characteristics:

- A defect and description process: To detect defects, inspectors always use user profiles and step through task scenarios. While stepping through the tasks, inspectors apply a task performance model and heuristics to learn where, in the user's task flow, to look for defects. Inspectors then describe the defects in a user-centered manner.

- An inspection team: Inspectors represent various knowledge domains, including software, hardware, documentation, support, and human factors engineering. Inspection team members also have a well-defined role during the inspection process.

- A structure within the usability lifecycle: Defect detection is framed within a structure of six steps: planning, kickoff meeting, preparation, logging meeting, rework, and follow-up.

Kahn and Prail (1994) noted that in many other inspection-type review sessions, team members do not spontaneously notice usability defects during scheduled activities. As a result, they provided some structure (e.g., forms, questions, and heuristics) to increase engineer recognition of defects. The intent was to help inspectors describe more effectively the defects found in the product and to aid in finding the best solution.

Kahn and Prail (1994) recommended giving inspectors a product description, user profiles, and task scenarios. The product description usually consists of screen drawings and explanatory text, but could consist of storyboards or a prototype. User profiles include a task identifier, user education, and user experience. Each task scenario includes the user's goal, the starting point in terms of the task situation and the product state, and any intermediary situations that the user will encounter. Kahn and Prail argue that task scenarios should not provide a list of task steps since, as part of the inspection, the inspectors will have to figure out how to perform the task. But it is important that inspectors do not work at too much of an abstract level, so as to overlook decisions that users will have to make. During the process, Kahn and Prail use a task performance model to help inspectors learn where, in the user's task flow, to look for defects. The model starts with a goal, followed by perceiving, planning, selecting, and acting. Each phase is associated with a short set of questions to help inspectors discover defects. Inspectors can use a checklist approach or priming approach when applying the task performance model. In the checklist approach, the inspector applies each appropriate phase of the model and, optionally, each appropriate task performance question. In the priming approach, the inspector examines the model and, optionally, the task performance questions, prior to stepping through the tasks in order to become sensitized or reacquainted with issues. The checklist approach would seem to find more defects but, for most inspections, would be too time-consuming and tedious.

Therefore, Kahn and Prail recommend using the priming approach if length of the evaluation is a concern.

The task orientation approach in the Kahn and Prail (Kahn & Prail, 1994) formal usability inspection method is important to the process. Their contention is that it would not be appropriate to do a usability inspection in a screen-by-screen manner, examining each field for what could go wrong, since this would not resemble the user's experience of doing the task. Instead, Kahn and Prail recommend that engineers should work in a step-by-step manner and examine each task step for what could go wrong.

Strengths of Formal Usability Inspections

Two major strengths are particularly noteworthy in the use of formal usability inspections. First, the method provides a structure and schedule to help the design team more effectively inspect a design product. Second, the method helps inspectors think about possible defects through the use of specific questions. Additionally, Kahn and Prail (1994) list some product-specific business benefits as a result of completing a formal usability inspection:

- The design team has a list of defects and implemented solutions,
- User testing will be more effective and efficient since users will encounter fewer problems,
- The design team has achieved a milestone in the user-centered lifecycle. In this way, the design team communicates to management that they are applying user-centered design, and
- The design team has user profiles and task scenarios. These can be reused furing future inspections and user testing.

Weaknesses of Formal Usability Inspections

The major weakness, especially for small product development efforts, is that formal usability inspections require several people with various domain backgrounds to complete the process (e.g., moderator, owner(s), inspectors, and scribe). A formal usability inspection is not a method designed for one person to use. In addition, although the process provides a structured approach for looking for defects, this approach does not provide a hierarchy of questions to lead the inspector to the specific problem. The questions used in formal usability inspections are at a very top level (e.g., "What problems could the user have when trying to physically perform the selected action?") with no sub-categories or branching structure to help the inspector continue thinking about the problem at a lower level. This method also does not help the inspector consider such important issues such as error recovery, exploration, and stacking of tasks.

Usability Walkthroughs

A usability walkthrough is a systematic review of a design on paper (Bias, 1991). Most usability walkthroughs involve a large group, with a usability professional as the session leader and facilitator. The group can consist of end users, product designers, documentation staff, and health/safety professionals in addition to the usability staff. In some areas, the group usability walkthrough is known as the "pluralistic usability walkthrough;" originally developed by Bias (1994) at IBM to bring together a team of representative user, product developers, and human factors professionals. Many usability walkthrough approaches have their roots in the structured walkthrough techniques originally described by Yourdon (1989). Usability walkthroughs share some characteristics with formal usability inspections and cognitive walkthroughs, but have two defining characteristics. First, usability walkthroughs do not ask evaluators to systematically consider the mental operations (such as goal formation) associated with system use (Karat & Bennett, 1991). Second, usability walkthroughs do not include a task performance model or a series of formal questions for the participants to consider. Rather, a panel of representative users and system designers step through tasks with a proposed system design in order to document usability issues along the way.

Strengths of Usability Walkthroughs

Usability walkthroughs provide immediate feedback and the increased buy-in achieved by having the developers present to hear the concerns of the representative users and human factors professionals (Bias, 1991; Bias, 1994). In addition, usability walkthroughs provide early performance and satisfaction data from users when a user interface prototype is not available. Even if a prototype is available, the usability walkthrough can be the ultimate in rapid test-redesign-retest usability engineering, with the just-derived new designs being discussed and evaluated in the walkthrough itself.

Weaknesses of Usability Walkthroughs

A primary weakness of the usability walkthrough is it must progress as slowly as the slowest person on the panel (Bias, 1994). Also, most participants find it hard to get an overall view of the flow when they are asked to methodically step through screen panels one at a time. As with formal usability inspections, the usability walkthrough cannot simulate all possible actions because time is generally limited for the review. Thus, exploration, browsing, and error recovery are not addressed in the usability walkthrough.

Heuristic Walkthroughs

Sears (1997) developed heuristic walkthroughs to combine several advantages of heuristic evaluation and the cognitive walkthrough. Sears incorporated a free-form evaluation with a list of usability heuristics from Nielsen's (1994a) heuristic technique and then added a list of user tasks and a list of questions that highlight important parts of the interaction process from the cognitive walkthrough. The result was a two-pass evaluation process where evaluators are guided by a prioritized list of user tasks, a list of usability heuristics, and a list of "thought-focusing" questions. In the first pass through the interaction design, evaluators explore tasks from a prioritized list. During the second pass, evaluators explore any aspect of the system they want while looking for usability problems.

Sears (1997) justifies using a task-based approach from evidence that a free-form evaluation, as is done in heuristic evaluation, results in finding a large number of minor usability problems (Nielsen, 1992). Ignoring user tasks, as noted by Sears, leads to the unfortunate side effect of identifying numerous false positives. However, Sears includes a free-form evaluation part to the heuristic walkthrough in order to help the evaluator find some of the less severe, but potentially important, problems. Sears includes a list of heuristics to help novices focus their attention on usability issues.

Strengths of Heuristic Walkthroughs

The primary strength of heuristic walkthroughs is the combination of important features from both the heuristic evaluation and cognitive walkthrough techniques. Since the heuristic walkthrough does not require evaluators to focus on complex cognitive issues, the training and evaluation time is reduced compared to the full cognitive walkthrough. Providing a list of heuristics helps evaluators focus their inspection of the interface.

Weaknesses of Heuristic Walkthroughs

Even though the heuristic walkthrough provides evaluators with a list of heuristics to focus their evaluation, the heuristics do not necessarily help the evaluators focus on the user interaction. Tasks, provided in the process, are intended to provide such a focus, but are less detailed than in the cognitive walkthrough. As a result, evaluators do not consider all potential cognitive issues related to the task. Evaluators that do not have the necessary cognitive psychology background generally have more difficulty exploring some of these important cognitive issues.

Model-Based Evaluations

Some researchers consider model-based evaluations to include cognitive modeling approaches such as GOMS (Card et al., 1983), Cognitive Complexity Theory (Kieras & Polson, 1985), and the CE+ Model (Polson & Lewis, 1990). However, these cognitive modeling approaches were not originally intended for inspection of user interaction design as discussed in this research. Rather, formal cognitive modeling approaches are intended for predicting expert users' performance on specific tasks with a given interface (Virzi, 1997). Formal cognitive modeling approaches do not lead directly to usability problem identification. As a result, cognitive modeling approaches are more useful for initially constraining the design space, answering specific design decisions, estimating the total time for task performance, providing the base from which both to calculate training time and to guide training documentation, and knowing which stages of activity take the longest time or produce the most errors (Olson & Olson, 1990).

One cognitive modeling approach that has found some usefulness in usability evaluation is Norman's (1986) theory of action model. Although not an evaluation method on its own, Norman's model provides researchers and practitioners an understandable process for examining problems in the context of user activity. Because of its basis in user interaction activities, several researchers have used Norman's model to help describe and explain problems identified using various methods. Two noteworthy efforts covered here include Cuomo and Bowen's (1994) stages of user activity analysis and Sutcliffe, Ryan, Springett, and Doubleday's (1996) Model-Mismatch Analysis method.

Stages of User Activity Analysis

Cuomo and Bowen (1992, 1994) used Norman's (1986) theory of action model to assess the usability of graphical, direct-manipulation interfaces with some success. Combining Norman's theory of action model with an encoding scheme for user action, these researchers were able to structure the integrated usability data, allowing easy extraction of important objective usability indicators such as number of actions per step, number of steps per task, and comprehensive error data. Cuomo and Bowen (1992) did not develop a new evaluation method, but rather used Norman's model to examine usability problems identified during an assessment of a graphical direct manipulation-style interface. Cuomo and Bowen's (1992) goal was to understand the types of usability problems found by three evaluation techniques. The three evaluation techniques included the cognitive walkthrough, heuristic evaluation, and the Smith and Mosier (1986) guidelines. Cuomo and Bowen (1992) wanted to learn whether the techniques identify problems across all stages of user activity and which are important to the usability of direct manipulation-style systems.

Results from the Cuomo and Bowen (1992, 1994) studies showed that the cognitive walkthrough method identifies issues almost exclusively within the action specification stage, while guidelines covered more stages. The cognitive walkthrough was best, however, and the guidelines worst at predicting problems that cause users noticeable difficulty (as observed during a usability study). All techniques were weak in identifying problems in the intention formation and evaluation stages of Norman's (1986) model. Guidelines identified the most problem types overall, followed by heuristic evaluation, with cognitive walkthrough finding the least.

An interesting result from the Cuomo and Bowen (1992, 1994) studies is the number of problem types confirmed by each stage of Norman's (1986) model. When summarizing their results, Cuomo and Bowen noted that the majority of problems were identified in the action specification stage (46%). Problems in the stages of execution (16%), interpretation (15%), and perception (13%) were next most frequent. Intention formation (7%) and evaluation (3%) were the stages with the least identified problem types.

Using Norman's (1986) theory of action model to classify types of usability problems shows great promise, especially in the usability testing environment as shown by Cuomo and Bowen (1994). Much more work is needed to modify Norman's theory of action model for use as an inspection technique early in the development of human-computer systems.

Strengths of Stages of User Activity Analysis

Cuomo and Bowen's (1992, 1994) use of Norman's (1986) theory of action model provides an understandable process for identifying the context of usability problems found during formative usability evaluation. The most useful part of the model is the ability to show practitioners where the majority of their usability problems occur in the user activity model. Such a view can potentially help developers focus on fixing problems in one part of the user activity model where users are having the most difficulty.

Weaknesses of Stages of User Activity Analysis

Norman's (1986) model, as used by Cuomo and Bowen (1992, 1994) in the evaluation of direct-manipulation interfaces, does not provide a direct method for finding usability problems. Norman's model can only be used after experts find problems using a specific UEM. Once experts find specific problems, the research team must provide additional resources to sort and classify problems according to the stages defined in Norman's model. Although such classification is potentially valuable, many practitioners may not have the time or resources to accomplish this post-analysis. An additional issue with Norman's model as used by Cuomo and Bowen is the emphasis on cognitive activities. Because Norman's model primarily focuses on cognitive activities, researchers and practitioners have some difficulty when trying to define and scope problems in the execution stage. Norman did not include a full description of problems that typically occur in this area, and rightfully so since his model was based on cognitive user activity research.

Model-Mismatch Analysis Method

Sutcliffe et al. (1996) combined Norman's (1986) theory of action model and Lewis et al.'s (1990) walkthrough methodology to form the Model-Mismatch Analysis (MMA) method. MMA was intended to provide comprehensive guidance for diagnosis of usability problems. MMA involves an analysis of observations of user-system interaction to discover causes of usability problems. The core feature of MMA is the development of phenotype observations and genotype causes to help diagnose usability errors. Phenotypes are observed manifestations of failure that can be classified and then linked to underling causes of failure (genotypes) and their attribution to either flaws in system design or human error. The analysis starts with observed critical incidents and breakdowns that are then placed in five phenotypes contexts: (1) start of task or subtask, (2) action selected or initiated, (3) during action execution, (4) action completed, and (5) action completed, task not complete. Springett (1998) modified the categories in the MMA method resulting in the following four phenotypes: (1) failure to find features, (2) rejection of a feature, (3) accidents/manipulation, and (4) unsatisfactory result. Both the work by Sutcliffe et al. (1996) and Springett provide a systematic way of initially placing observations into top-level categories derived from Norman's theory of action model.

Once observations are classified by phenotypes, the MMA method continues with mapping phenotypes to possible explanations for each problem (i.e., genotypes). Possible explanations are explored through a walkthrough process leading to heuristics the evaluator can select as the most likely cause (e.g., inadequate feedback, mode error, missing functionality, etc.). Sutcliffe et al. (1996) use Norman's model with an error correction and non-goal directed exploration sub-cycles to fully exploit possible explanations for phenotype observations.

An additional feature of the MMA method is a user task model analysis. Users are asked to describe the steps they would expect to perform manually and then recall the evaluation task from memory. The data from the user task model analysis is used to ascertain the extent of task compatibility mismatches. Sutcliffe et al. (1996) suggest that systems provide better task support when the users' model is compatible with the original system model. In addition, Sutcliffe et al. use the user task model analysis to discover if system actions are being learned.

Springett (1998) summarized results from his error classification study with somewhat different findings than the Cuomo and Bowen (1992, 1994) studies. For example, Springett found the most common phenotype was unsatisfactory result of a task-action (52%). In the Cuomo and Bowen studies, problems with unsatisfactory results (i.e., evaluation stage) ranked last in frequency. The next most common phenotype in the Springett study was the inability to find feature (24%); a category most closely resembling the frequently occurring action specification problems in the Cuomo and Bowen studies.

Strengths of Model-Mismatch Analysis

The primary strength of the MMA method is the diagnosis of errors found during usability evaluation. Although some errors do not require a deep analysis, a number of usability problems require further analysis in order to understand the context of the underlying cause. The MMA method, with its phenotype and genotype classification process, appears to provide useful categorization that can inform evaluators about the type of design modification required.

Weaknesses of Model-Mismatch Analysis

As with other efforts using Norman's (1986) model (i.e., Cuomo & Bowen, 1992; Cuomo & Bowen, 1994), the MMA method does not provide a direct method for finding usability problems. MMA is only useful once problems are identified through a specific UEM. Although useful, the user task model analysis feature of the MMA method is quite time-consuming and may not be appropriate for most evaluation effort.

EVALUATING THE EFFECTIVENESS OF USABILITY EVALUATION METHODS

Among interactive system developers and users there is now much agreement that usability is an important part of software systems. However, as discussed in the previous section, many ways exist to evaluate the usability of an interaction design (i.e., many UEMs), with much room for disagreement and discussion about the relative merits of the various UEMs. As more new methods are being introduced, the variety of alternative approaches and a general lack of understanding of the capabilities and limitations of each has intensified the need for practitioners and others to determine which methods are more effective, and in what ways and for what purposes. In reality, researchers find it difficult to reliably compare UEMs because of three shortcomings:

- The HCI literature has not established standard criteria for comparison,
- Definitions, measures, and metrics are not standardized in a way that is useful for researchers and practitioners, and
- No stable, standard processes currently exist for UEM evaluation and comparison.

Lund (1998) noted the need for a standardized set of usability metrics, citing the difficulty in comparing various UEMs and measures of usability effectiveness. As Lund points out, no single standard exists for direct comparison, resulting in a multiplicity of different measures used in the studies, capturing different data defined in different ways. Consequently, very few studies clearly identify the target criteria against which to measure success of a UEM being examined. As a result, the body of literature reporting UEM comparison studies does not support accurate or meaningful assessment or comparisons among UEMs. For example, Gray and Salzman (1998) recently documented specific validity concerns about five popular UEM comparison studies. A key concern noted by Gray and Salzman is the issue of using the right measure (or measures) to compare UEMs in terms of effectiveness.

Before developing experimental designs to test new evaluation methods, researchers need to define and explain the characteristics and metrics of effectiveness so that developers can look at usability evaluation results and make relevant conclusions. The review in this section highlights some of the specific challenges that researchers and practitioners face when comparing UEMs and provides some techniques for selecting criteria and reliably measuring effectiveness.

Criteria Selection

To evaluate the effectiveness of a UEM, and especially to compare the effectiveness of UEMs, usability researchers must establish a definition for effectiveness and a criterion, or criteria, for evaluating and comparing it. The criteria are stated in terms of one or more performance-related (UEM performance, not user performance) measures (effectiveness indicators), which are computed from raw empirical usability data (e.g., usability problem lists) yielded by each UEM. Making the right choice for criteria and performance measures depends on understanding the alternatives available and the limitations of each.

The selection of criteria to evaluate a UEM is not essentially different from criteria selection for evaluation of other kinds of systems (Meister, Andre, & Aretz, 1997). In the evaluation of large-scale systems such as military weapon systems, for example, customers (e.g., the military commanders) establish *ultimate* criteria for a system in the real world. Ultimate criteria are usually simple and direct – for example, that a certain weapon system will win a battle under specified conditions. Military commanders cannot measure such ultimate criteria directly outside of an actual combat environment. As a result, military commanders establish specific other attributes, called *actual* criteria, which are more easily measured and which are believed to be effective predictors of the ultimate criteria. To illustrate, commanders might establish the following characteristics as actual criteria for military aircraft performance: aircraft must fly at X thousand feet, move at Y mach speed, and shoot with Z accuracy. As actual criteria, these measures are only indicators or predictors of the ultimate criterion and are more valuable as predictors if they can be validated, which can happen only when real combat clashes occur.

Measures to be used in actual criteria represent operational parameters that can be computed by consistent means that are agreed upon and understood in the same way. If system reliability was a goal, for example, mean-time-between-failure (MTBF), is a good measure because everyone understands what it means and how to compute it. HCI researchers do not have any measures this standardized yet in usability, so they define their own measures for each study. To be useful and repeatable in an actual criterion, a measure must have at least these characteristics:

- a solid definition, understandable by all,
- a metric, to be computed from raw usability data,
- a standard way to measure, take data, and
- one or more levels of performance that can be taken as a "score" to indicate "goodness."

The degree to which actual criteria are successful predictors of ultimate criteria is the essence of the concept called *criterion relevance*, illustrated by the intersection of the two circles in Figure 2-1. If stealth technology makes it unnecessary to fly at 80,000 feet, then the altitude criterion is no longer a useful predictor of the ultimate criterion causing that part of the actual criterion to fall outside the intersection with the ultimate criterion. Because this part of the actual criterion contaminates the approximation to the ultimate criterion, it is called *criterion contamination.*

If military commanders leave out an important measure that should be included in the estimate of ultimate criterion, the actual criterion is deficient in representing the ultimate criterion and the part of the ultimate criterion not represented falls outside the intersection in the part called *criterion deficiency.*

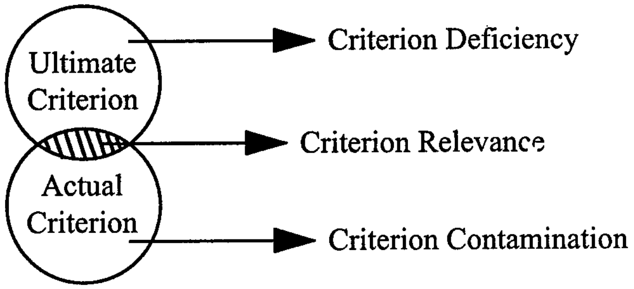


FIGURE 2-1. Relationship Between Ultimate and Actual Criteria.

Although military weapon systems and UEMs are quite different, the relationship between ultimate and actual criteria is still applicable. The ultimate criterion for UEMs is how well they help inspectors discover problems that impact users in real work contexts. When a researcher only focuses on one measure, the chances are great that this one characteristic is deficient in terms of measuring overall effectiveness of the UEM. Alternatively, when a researcher looks at several measures, the criteria may be contaminated if the UEM finds lots of problems, but very few of them actually relate to problems encountered by real users. Therefore, the goal in effectiveness studies of UEMs should be to find those measures that relate the actual criteria to the ultimate criteria.

Goldstein (1993) specifically addresses deficiency and contamination problems when evaluating the effectiveness of training programs. According to Goldstein, criterion deficiency can be reduced through the use of a composite measure (i.e., group of measures combined into a single measure) or a multiple-criterion approach (i.e., evaluating many independent measures). Current UEM studies look mostly at the multiple-criterion approach; however, researchers may find some value in establishing weightings for sub-criteria of effectiveness in order to use a composite measure.

On the issue of contamination, Goldstein (1993) notes the existence of criterion contamination can lead to incorrect conclusions regarding the validity of training programs and identifies several biases that can contribute to contamination (e.g., treating individuals from various instructional techniques differently). Applying the same reasoning to UEM studies, criterion contamination may be due to two primary factors: (1) how the UEM is designed to find problems, and (2) how well the individual uses a particular UEM. Most research studies have only focused on optimizing the second factor (e.g., using experts), without addressing the characteristics of the UEM itself. More work is needed to modify our current methods to control these two sources of biases in order to reduce criterion contamination.

Ultimate Criteria for UEM Effectiveness

Criterion relevance applies to UEMs as well as military systems. In a somewhat analogous way to the simple ultimate criterion used in the case of a military system, the ultimate criterion for UEM evaluation and comparison translates to: How well does the UEM help inspectors discover real usability problems?

To the extent that any practical means for determining realness in the actual criterion will result in some errors, there will be both criterion contamination and criterion deficiencies. But once the actual criterion (including the method for determining realness) is established, those issues essentially disappear from immediate consideration and the focus is on the actual criterion as the standard: How well does the UEM help inspectors discover "real" (as determined by the method of the actual criterion) usability problems?

Designing Realistic Actual Criteria for UEM Effectiveness

The choice of both ultimate criteria and actual criteria depends on *goals* for doing evaluation in the first place (Scriven, 1967). The actual criterion is central to UEM evaluation or comparison and its design is crucial to the success of any such study. If a UEM study is to be used to compare a UEM against a standard to answer the question "How good is the UEM?," the question faced within the study is: How well does the UEM meet the actual criterion? If a study is designed to compare one UEM against another to answer the question "Which UEM is better?," the real question of the study is: Which UEM meets the actual criterion better? Both kinds of comparison questions require actual criteria to address two important issues:

- Which usability problems are real?
- Which UEM performance comparison measures to use?

Which Usability Problems are Real?

A usability problem (e.g., found by a UEM) is real if it is a predictor of a problem that users will encounter in real work-context usage and that will have an impact on usability (user performance, productivity, and/or satisfaction). This approach would exclude problems with trivially low impact and situations real users would/could not encounter. The emphasis on real users is important in this definition, because many of the UEMs evaluated in studies are usability inspection methods, where the inspectors encounter problems that do not always predict usability problems for real users. In any case, this definition of realness belongs more to ultimate criteria than to any actual criterion, since it does not yet offer an operational way to test for the stated conditions. Two practical ways to determine realness include:

- Expert judgment of each candidate usability problem as to whether it qualifies as real, and
- Comparison against a standard list of usability problems known to be (or to approximate) "the real usability problems" of the target interaction design.

Determining Realness by Expert Review and Judgment

To determine the realness of usability problems by review and judgment of expert(s), one or more usability experts examine each candidate usability problem and determine if the problem is real or not. This technique can also have the effect of accumulating a standard list, if the judgment results can be saved and reused. This technique can also be combined with other techniques discussed below to filter the "standard" usability problem lists, ensuring that the results are, by expert judgment, real.

Often designers of UEM studies find the guidelines for realness to be used in expert judgment are too vague or general to be applied reliably and the judgments can vary with the expert and other experimental conditions. The variance associated with an expert review introduces the possibility of a bias causing the usability problem lists of each UEM to be judged differently. As an alternative, the experimenters seek a "standard" usability problem list as a single standard against which to compare each UEM's output.

Determining Realness by Comparing with a Standard Usability Problem List

If an evaluator had a complete list of all the real usability problems that exist in a given target interaction design, that evaluator could ascertain the realness of candidate usability problems found by a UEM by determining whether each such usability problem is in the standard list. However that determination is expressed, it will involve an underlying comparison of a candidate usability problem with each usability problem in the standard list. The comparison approach for determining realness includes the following three approaches:

- Describing usability problems as elements of sets,
- Computing a standard usability problem set, and
- Comparing a UEM's output with the standard.

Describing usability problems as elements of sets. In practice, each UEM produces a list of usability problems. Comparison of UEMs requires comparison and manipulation of their usability problem lists. It is useful to think of each UEM as producing a *set* of usability problems, because it allows for thinking of the list as unordered and allows formal expressions of important questions. Cockton and Lavery (1999) favor this same choice of terminology for much the same reasons. For example, a researcher might need to ask whether a given UEM finds a certain known problem in a target design. On the other hand, the researcher might need to know what usability problems the outputs of UEM_1 and UEM_2 have in common, or what results when merging the outputs of UEM_1 and UEM_2. These questions are about set membership, set intersections, and set unions. Viewing usability problem lists as sets also affords simple set operations such as union, intersection, and set difference to manipulate the usability problems and combine them in various ways to calculate UEM performance measures.

Computing a standard usability problem set. The second technique for determining realness requires the experimenters to establish a "standard" UEM that can be used to generate, as a comparison standard, a touchstone set of usability problems deemed to be "the real usability problems" existing in the target interaction design of the study. This standard usability problem set will be used as a basis for computing various performance measures as parts of actual criteria. The touchstone set is part of an actual criterion because it can only approximate the theoretical ultimate real usability problem set, a set that cannot be computed. Some of the possible ways to produce a standard-of-comparison usability problem set for a given target interaction design include:

- Seeding with known usability problems,
- Lab-based usability testing,
- Asymptotic lab-based testing, and
- Union of usability problem sets over UEMs being compared.

Sometimes experimenters will "seed" or "salt" a target system with known usability problems, an approach that can seem attractive because it gives control over the criterion. In fact, seeding with known usability problems is one of the few ways the experimenters can know about all the existing problems (assuming there are no real problems in the system before the seeding). But many UEM researchers believe salting the target system is not a good basis for the science of a UEM study because the outcome depends heavily on experimenter skill (in the salting), putting ecological validity in doubt. Experienced usability practitioners will know that contrived data can seldom match the variability, surprises, and realness of real usability data from a real usability lab.

Traditional lab-based usability testing is the de facto standard, or the "gold standard" (Landauer, 1995), used most often in studies of UEM performance. Lab-based testing is a UEM that produces high-quality, but expensive, usability problem sets. Often lab-based UEM performance is unquestioned and thought of as a good actual criterion, suitable for use as a standard of comparison to evaluate other UEMs. Because a lab-based usability test is such a well-established comparison standard, it might be thought of as an ultimate criterion, especially when compared to usability inspection methods. However, a lab-based usability test does not meet the definition for an ultimate criterion. In the usability lab, users are constrained and controlled. Developers decide which tasks users should perform and what their work environment will be like (usually just the lab itself). Some researchers and practitioners would like more data on how well lab-based testing is predictive of real usability problems and under what conditions it best plays this role, but it is difficult to find an experimental standard good enough to make that comparison.

Despite these possible deviations from the ultimate, the experience of the usability community with lab-based testing as a mainstream UEM for formative evaluation within the interaction development process has led to a high level of confidence in this UEM. Other UEMs have arisen, not because of a search for higher quality, but mostly out of a need for lower cost alternatives. In any case, the typical lab-based usability test employs several users as subjects along with one or more observers and produces a union of problems found by all users. Given that some usability problems, even from lab-based testing, can be of questionable realness, it is best to combine the lab test with expert review to eliminate some of the problems considered not real, thus improving the quality of the usability problem set to be used as the actual criterion.

The typical usability lab test will miss some usability problems. In fact, most lab tests are deliberately designed with an objective of cost-effectiveness, at an acknowledged penalty of missing some usability problems. Virzi (1992) has investigated sample size considerations for usability evaluation studies and found average detection rates ranging from 0.32 to 0.42. Using the formula $1 - (1 - p)^n$, researchers have shown that five evaluators (n) find approximately 80% of the usability problems in a system if the average detection rate (p) is at least 0.30 (Nielsen, 1994b; Virzi, 1990; Virzi, 1992; Wright & Monk, 1991). However, average detection rates can be as low as 0.16 in office applications as shown by Lewis (1994). Figure 2-2 shows the problem discovery likelihood when individual detection rates range between a low of 0.15 and a high of 0.45 using the formula $1 - (1 - p)^n$. The graph in Figure 2-2 clearly shows that the asymptote of problem detection varies significantly depending on the individual detection rate. Only 8 evaluators are needed to find 95% of the problems when the detection rate is 0.45, while as many as 19 evaluators are needed to find the same amount when the detection rate is 0.15.

Since the total usability problems found levels off as the number of users increases, the asymptotic level could be thought of as a good approximation to the level of the ultimate criterion (after any non-real problems are removed). Thus, extending the usual lab-based usability test to include several more users is a good, but expensive, choice for producing a "standard" usability problem set from the target design as part of an actual criterion.

Another technique often used to produce a standard usability problem set as a criterion for being real is the union set of all the individual usability problem sets, as found by each of the methods being compared (Sears, 1997). This method has a very serious drawback in that it eliminates the possibility to consider validity as a UEM measure, because the basis for metrics is not independent of the data.

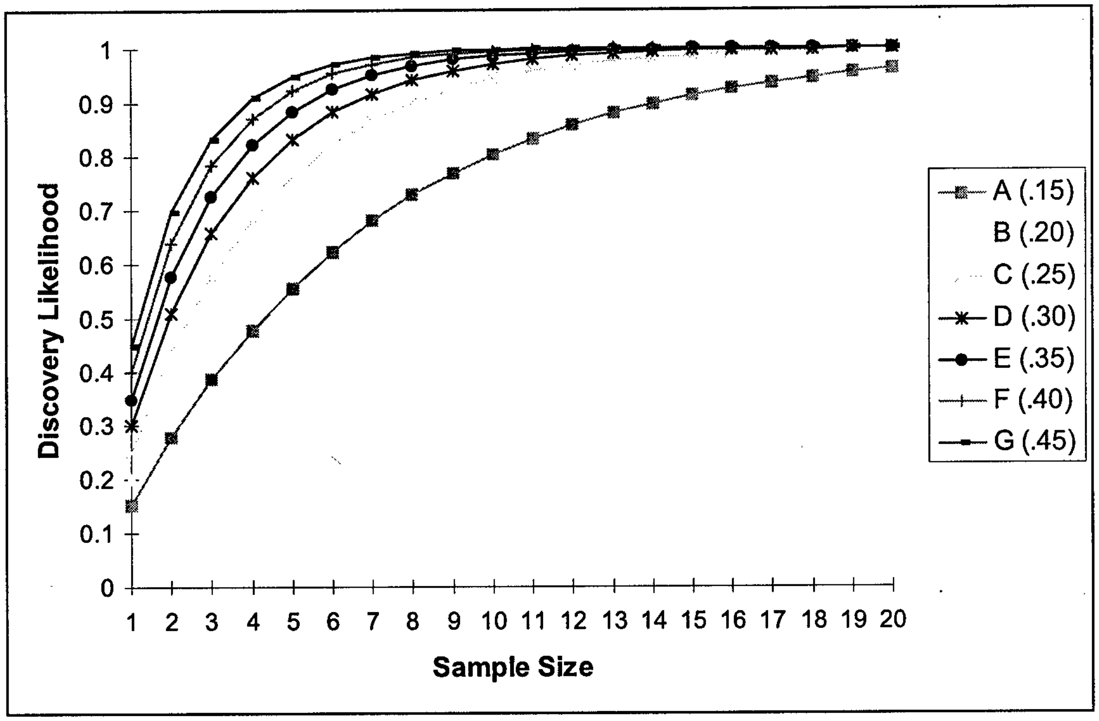


FIGURE 2-2. Predicted Problem Discovery Likelihood (adapted from Lewis, 1994).

Comparing Usability Problem Descriptions

Gray and Salzman (1998) correctly criticize just counting usability problems for UEM measures, without determining if some usability problems found overlap or duplicate others. A determination of overlap cannot be made, though, without an ability to compare usability problem descriptions. Determining realness by comparing with a standard usability problem set also requires comparison. Comparison requires complete, unambiguous usability problem descriptions that facilitate distinguishing different types of usability problems.

This comparison is straightforward in abstract sets, where each element is unambiguously identified by name or value. If x ∈ A and x ∈ B, then the appearance of x in A is identical to its appearance in B. However, usability problem sets from UEMs are more difficult to compare because they involve enumerated sets in which elements are represented by narrative problem descriptions and elements, not having a unique canonical identity.

Because usability problem descriptions are usually written in an *ad hoc* manner, expressed in whatever terms seemed salient to the evaluator at the time the problem is observed, it is not unusual for two observers to write substantially different descriptions of the same problem. Thus, in order to perform set operations on usability problem sets, one needs the ability to determine when two different usability problem descriptions are referring to the same underlying usability problem. This kind of comparison of textual problem descriptions is usually done by expert judgment, but is subject to much variability. Researchers need a standard way to describe usability problems and a framework within which usability problem descriptions can be more easily and more directly compared. One proposed method for standardizing usability problem descriptions is discussed in Chapter 3.

Designing Actual Criteria – Which UEM Performance Measures to Use?

Bastien and Scapin (1995) identified three measures for examining an evaluation method: validity, thoroughness, and reliability. Sears (1997) also points out these same measures, giving them somewhat different operational definitions. These three basic measures are:

- Thoroughness: Evaluators want results to be complete; they want UEMs to find as many of the existing usability problems as possible.
- Validity: Evaluators want results to be "correct;" they want UEMs to find only problems that are real.
- Reliability: Evaluators want consistent UEM results, independent of the individual performing the usability evaluation.

In addition to these measures, a metric known as *effectiveness*, can be defined in terms of the combination of thoroughness and validity (i.e., optimizing both). Practitioners who must get real usefulness within tightly constrained budgets and schedules would benefit from metrics such as *cost effectiveness* and *downstream utility*.

As Gray and Salzman (1998) point out, multi-measure criteria are needed, not just one-dimensional evaluations. When a researcher only focuses on one measure (e.g., thoroughness), it is unlikely that this one characteristic will reflect overall effectiveness of the UEM. For example, if an inspection method focuses on high reliability, it does not guarantee that the output of an inspection will produce quality problem reports that communicate problems and causes precisely and suggest solutions for down-stream redesign activities. In addition to thoroughness and validity, researchers may also be interested in reliability, cost effectiveness, downstream utility, and usability of UEMs. Any of these issues could form the criteria by which researchers judge effectiveness. Although it is nearly impossible to maximize all of the parameters simultaneously, practitioners must be aware that focusing on only one issue at the expense of others can lead to an actual criterion having significant criterion deficiency.

The main goal addressed by UEM evaluation is to determine which UEM is "best." Beyond that, researchers ask, "Best for what?" Ultimate criteria should be selected with this more specific question in mind. In effectiveness studies of UEMs, the objective should then be to find those measures comprising actual criteria to best relate them to the ultimate criteria. Thus, the measures are a way of quantifying the question of how well a UEM meets the actual criteria. Even when researchers combine several measures, they still need to be aware of contamination issues. For example, the criteria may be contaminated if the UEM finds lots of problems, but very few of them are real as defined in terms of being good predictors of problems users will actually encounter.

Thoroughness

Thoroughness is perhaps the most attractive measure for evaluating UEMs. Sears (1997) defines thoroughness as a measure indicating the proportion of real problems found using a UEM to the real problems existing in the target interaction design:

$$\mathit{Thoroughness} = \frac{\text{number of real problems found}}{\text{number of real problems that exist}} \qquad (1)$$

For example, if a given UEM identified 30 problems, with only 10 coming from a set of 20 real usability problems determined to be in a target system, that UEM would be said to have yielded a thoroughness of 10/20 = 0.50. UEMs with low thoroughness waste developer resources by leaving important usability problems unattended after investment in the usability evaluation process.

How to calculate the denominator is often not clearly defined in most UEM studies. The denominator in this equation provided by Sears (1997) is essentially about some theoretical data, or ultimate criterion. For example, the number of real problems that exist in a system can only be determined when the system is deployed and data collected on every user. Most researchers would find that collecting such data to be nearly impossible for systems used in comparison studies. Thus, researchers usually only approximate the denominator in this case by using a method (or methods) to identify potential problems. Whatever method is used to determine realness, that method can also be considered a UEM, in this case a definitional UEM$_A$ (A referring to "actual criteria") that, however arbitrarily, determines realness. The output of this, so far undefined, UEM is considered the "perfect" yardstick against which other UEMs are compared. When applied to the target interaction design, UEM$_A$ produces a definitional usability problem set, A, defining those real problems that exist in the design. If P is the set of usability problems detected by some UEM$_P$, then the numerator of thoroughness for UEM$_P$ is computed by an intersection as in the following equation:

$$Thoroughness = \frac{|\mathrm{P} \cap \mathrm{A}|}{|\mathrm{A}|} = \frac{|\mathrm{P'}|}{|\mathrm{A}|} \qquad (2)$$

where P' is the set of *real* usability problems found by UEM$_P$. In the earlier example described above, the 20 real problems represents A, and P ∩ A are the 10 problems common to both sets.

Researchers have often not been clear in their own definition and use of the thoroughness measure. For example, the wording used for the numerator and denominator in the Sears definition of thoroughness leaves plenty of room for misinterpretation. In some cases, researchers are not able to use a lab-based usability test as a standard comparison because of resource constraints or end-user availability. Instead, researchers often resort to calculating the denominator by combining the union of usability problems from various methods used in the study. However, not all researchers would agree that using the union of three methods is an appropriate actual criterion in replace of formal lab-based usability testing. In addition, the union of usability problems from various methods presents a problem when calculating validity as discussed in the next section.

Validity

In general terms, validity is a measure of how well a method does what it is intended to do. According to Sears (1997), a technique is valid if evaluators are capable of focusing on relevant issues. Validity accounts for the fact that evaluators are known to identify a certain amount of problems that are not relevant or important. Validity allows the researcher to focus on a particular issue and measure how much extra effort is being spent on issues that are not important. In equation form, Sears defines validity as a measure indicating the proportion of problems found by a UEM that are real usability problems:

$$\mathit{Validity} = \frac{\text{number of real problems found}}{\text{number of issues identified as problems}} \qquad (3)$$

Validity and thoroughness can be computed using the same data, the usability problem sets generated, and the realness criterion. Consider the same example described for thoroughness. For a given UEM that finds 30 usability problems in a target system, of which only 10 were real, the validity rating would be 10/30 = 0.33. UEMs with low validity find large numbers of problems that are not relevant or real, obscuring those problems developers should attend and wasting developer evaluation, reporting, and analysis time and effort.

As with thoroughness, computing validity in terms of sets, gives the following equation:

$$\mathit{Validity} = \frac{|\mathrm{P} \cap \mathrm{A}|}{|\mathrm{P}|} = \frac{|\mathrm{P}'|}{|\mathrm{P}|} \qquad (4)$$

One approach researchers might use for determining the definitional usability problem set, A, is a lab-based usability test. When compared against a lab-based usability test, validity measures the ability of a particular method to find usability problems that users potentially will have with the system. For example, a researcher interested in finding out if a particular method identifies problems related to actual user problems might use a lab-based test with users to collect a set of important problems. The extent that a particular method is able to identify these important problems reflects its validity.

If the union of usability problems from various methods is used to produce the standard usability problem set, A, the results are flawed. For example, building a union set from a set of individual usability problem sets involve the following steps:

1. Look at each candidate problem in each individual set, one at a time.
2. Determine if candidate problem is "real" (per criterion) and discard if not real.
3. If real, compare candidate problem to each of those in union set.
4. If candidate is represented in set (already in union set in some form), possibly use some of the wording of candidate problem to refine or improve wording of the problem that represents the candidate in the union set; discard candidate.
5. If candidate is not represented in union set, add to set.

This technique works better if the number of methods being compared is relatively large, increasing confidence that at least one of the methods finds almost all the real problems. One negative effect of this approach is to eliminate validity as a metric. This effect is important enough to all but preclude the union of usability problem sets as a viable approach.

As an example, suppose a UEM comparison study compared UEM_P, UEM_Q, and UEM_R. Let P(X) be the usability problem set found in interaction design X by UEM_P, and so on for UEM_Q and UEM_R. No standard UEM, UEM_A, is available, so the union of the output sets of the UEMs being evaluated is used as a substitute for the output of the standard, UEM_A:

$$A(X) = P(X) \cup Q(X) \cup R(X) \tag{5}$$

Even though most research studies do not say so explicitly, they are using this union as the basis of an actual criterion. The number of usability problems in this union is bounded by the sum of cardinalities of the participating usability problem sets:

$$|A(X)| = |P(X) \cup Q(X) \cup R(X)| =$$

$$|P(X)| + |Q(X)| + |R(X)| \;-\; |P(X) \cap Q(X) \cap R(X)| \tag{6}$$

The more UEMs participating in the comparison, the more real usability problems will be included in the union usability problem set. Unfortunately, more non-real problems are also likely to be included, decreasing validity, but this approach to an actual criterion, by definition, prevents any possibility of detecting the reduced validity. For example,

$$\textit{Validity of } UEM_P = \frac{|\,P\,(X) \cap A\,(X)\,|}{|\,P\,(X)\,|} \tag{7}$$

Because A(X) is a union containing P(X), P(X) is a proper subset of A(X) and nothing is removed from P(X) when it is intersected with A(X). This approach guarantees that the intersection of the UEM usability problem set and the standard usability problem set (the union) will always be the UEM usability problem set itself. This means that all usability problems detected by each method are always real and validity is 100% for all participating methods. In other words:

$$\textit{Validity of } UEM_P \;=\; |P(X)| \,/\, |P(X)|\text{, which is identically equal to 1.0} \tag{8}$$

Effectiveness

Thoroughness and validity have rather complete analogies to the concepts of recall and precision in the field of information storage and retrieval, terms that refer to measures of retrieval performance of an information system from a target document collection (Salton & McGill, 1983). The document collection searched by an information system corresponds to the target interaction design being evaluated and the information system corresponds to the UEM. Precision and recall are based on a concept called relevance (reflecting a determination of relevance of a document to a query), analogous to the concept of realness in UEMs. Relevance is the criterion for computing precision and recall.

Recall corresponds to thoroughness and is a measure indicating the proportion of relevant documents (for example) found in a collection by an information system to the total relevant documents existing in the target document collection.

$$Recall = \frac{\text{number of relevant documents found}}{\text{number of relevant documents that exist}} \qquad (9)$$

Precision is the proportion of the documents retrieved by an information system that are relevant:

$$Precision = \frac{\text{number of relevant documents found}}{\text{total number of documents retrieved}} \qquad (10)$$

Just as neither precision nor recall alone is sufficient to determine information system retrieval effectiveness, neither thoroughness nor validity alone is sufficient for UEM effectiveness. For example, high thoroughness alone allows for inclusion of problems that are not real, and high validity alone allows real problems to be missed. Following the concept of a figure of merit in information retrieval, the "figure of merit" for UEM effectiveness can be defined as the product of thoroughness and validity:

$$\text{Effectiveness} = \text{Thoroughness} \times \text{Validity} \qquad (11)$$

Effectiveness can range from 0 to 1; reflecting the input from thoroughness and validity. Where either thoroughness or validity is low, effectiveness will be low also.

Figure 2-3 shows a theoretical relationship among thoroughness, validity, and effectiveness. The curve shows a tradeoff between thoroughness and validity. As a UEM design is adjusted to be more sensitive to possible usability problems, the number of candidate usability problems identified increases, increasing both real and non-real problems identified. As this adjustment approaches near-perfect thoroughness at the right-hand side, validity will be dropping toward zero due to inclusion of more non-real usability problems in the process of capturing most of the real problems.

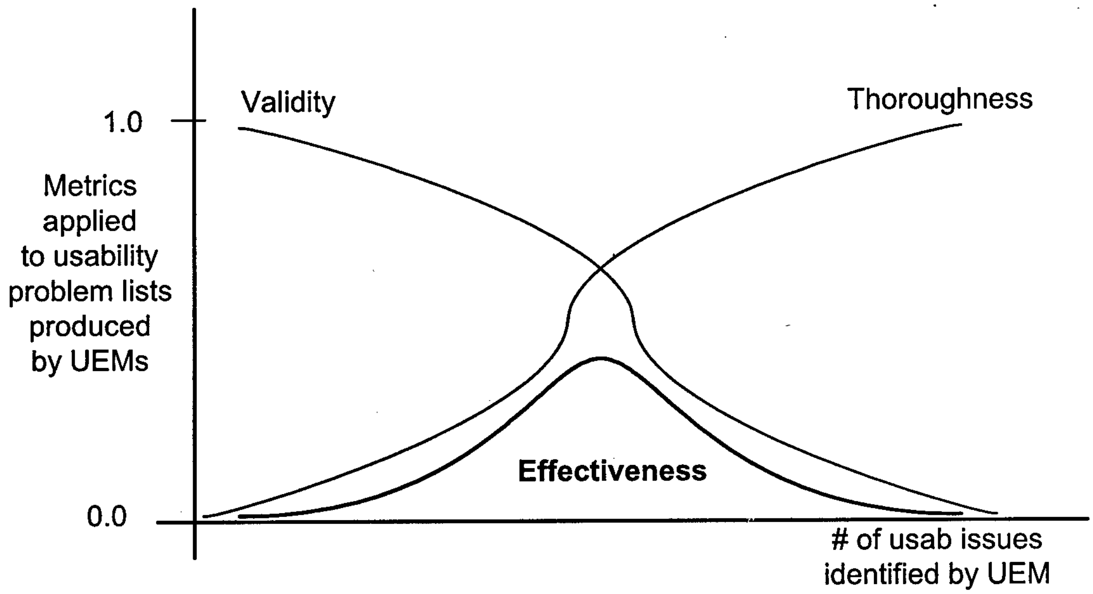


FIGURE 2-3. Conjecture About Relationship of Thoroughness, Validity, and Effectiveness

A developer who tunes a UEM design simply by adjusting its sensitivity, moving its operating point on the abscissa of the graph, in order to improve thoroughness will likely pay a price in reduced validity, and vice versa. The effectiveness measure offers a compromise target for optimization. There can also be more interesting UEM design factors, beyond sensitivity, that might be specific to one measure without a negative effect on the other.

Some researchers (e.g., Gray & Salzman, 1998) have alluded to the concepts of hits, misses, false alarms, and correct rejections in the context of UEM outputs. These concepts originated with hypothesis testing error types explained in most modern books on statistics or experimental research and adapted for signal detection theory (Egan, 1975; Swets, 1964). Further adapting this terminology to usability problem detection provides four cases, as shown in Figure 2-4, to describe the accuracy of a UEM with respect to the realness of the problems it detects, as determined by some actual criterion, A.

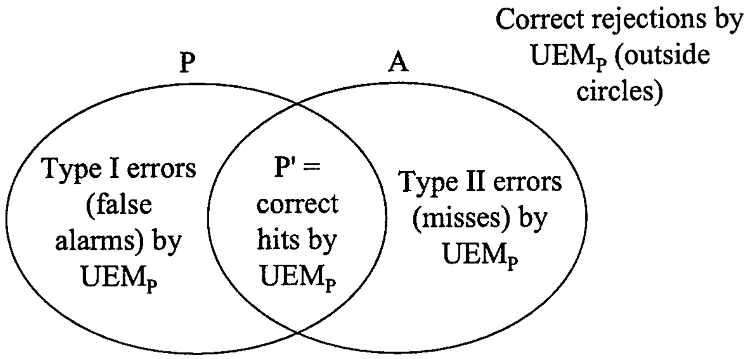

Correct rejections by UEM$_P$ (outside circles)
P
A
Type I errors (false alarms) by UEM$_P$
P' = correct hits by UEM$_P$
Type II errors (misses) by UEM$_P$

FIGURE 2-4. Venn Diagram Comparing Usability Problem Set Against Actual Criterion Set.

High thoroughness is achieved in a UEM by realizing most of the correct hits and by avoiding most of the Type II errors (misses). Similarly, high validity of a UEM derives from avoiding most of the Type I errors (false alarms) and realizing most of the correct rejections. False alarms do not affect thoroughness, but do detract from validity. The intersection in the center and the area outside both ovals represents the areas where the UEM is in agreement with the actual criterion. The same diagrammatic technique could be used to compare usability problem sets of two different UEMs, where the left hand oval is the usability problem set, P, of UEM$_P$ and the right hand oval is the set Q of method UEM$_Q$.

An important issue for both thoroughness and validity is how well the output of a UEM matches the "truth" about real problems existing in a given target system. Although some researchers may argue that it is impossible to obtain the "truth" about real problems, the important issue is really about isolating the actual criteria with respect to the ultimate criteria. Once a suitable actual criterion is established, it stands in stead of the ultimate criterion ("truth") and becomes the standard for determining realness of usability problems for the study. The study can be performed without the need to look beyond the actual criterion for truth. If a better approximation can later be found to the truth of the ultimate criterion, the actual criterion can be updated and studies repeated, again without concern about truth within the study.

Reliability

Reliability implies that similar results should be obtained under similar conditions. Sears (1997) measures reliability by using the ratio of the standard deviation of the number of problems found to the average number of problems found.

$$
\begin{aligned}
&\mathit{Reliability}_{temp} = 1 - \frac{\text{stdev (\# Real Problems Found)}}{\text{average (\# Real Problems Found)}} \\
&\mathit{Reliability} = \text{Maximum}(0, \text{R}_{\text{temp}})
\end{aligned}
\qquad (12)
$$

Sears (1997) proposes using this two-step process for computing reliability to eliminate the remote possibility of negative values for R_{temp}. Thus, reliability, as measured by Sears determines whether different evaluators, or groups of evaluators, tend to find similar numbers of problems when applying a given usability evaluation method. Unfortunately, using the reliability definition provided by Sears, a particular method might have high reliability (number of problems), yet still show inconsistency in terms of the types of problems identified by inspectors. Thus, just having reliability calculated statistically does not go very far at answering questions about agreement. Sears suggests that a more relevant reliability measure might focus on the specific problems identified by each evaluator, rather than just the number of problems found.

Reliability is often not included in comparison studies because researchers and practitioners desire some level of variance between evaluators. That is, if evaluators using a particular method find the same list of 10 problems, the reliability score is 1.0, but this method is most likely not very thorough at finding a great percentage of the problems. What is more important to researchers is the consistency of different evaluators when using a given method, especially when attempting to label and describe a problem. Such consistency is lacking in inspections methods such as the heuristic evaluation and cognitive walkthrough. Results are too dependent on evaluator expertise and generally result in low reliability numbers.

When inspection methods have a taxonomy or framework that is intended to help find and label usability problems, it is important to find out how reliable evaluators are at ending up in the same place in the framework. As a formal measure, reliability is an index of agreement between two or more sets of nominal identification, classification, rating, or ranking data. In this sense, reliability is a measure of observer agreement. Although several methods are available to measure reliability (Meister, 1985), Cohen's kappa (Cohen, 1960) is commonly used to examine observer agreement for categorical lists or taxonomies, especially when chance agreement is a consideration. Kappa (κ) is a measure of the proportion of agreement beyond what would be expected on the basis of chance and is calculated using the following equation from Cohen:

$$\kappa = \frac{p_o - p_c}{1 - p_c} \qquad (13)$$

where p_o is the proportion of observed agreement and p_c is the proportion of agreement expected by chance.

Severity Ratings

While a binary test of problem impact is necessary (i.e., a problem is real or it is not), it is not sufficient. A usability problem judged to be real can still have either only a small impact on user satisfaction or it might have a show-stopping impact on user task performance.

To further discriminate among degrees of impact, practitioners have extended the binary concept of realness into a range of possibilities called severity levels. Severity thus becomes another measure of the quality of each usability problem found by a UEM, offering a guide for practitioners in deciding which usability problems are most important to fix. The working assumption is that high severity usability problems are more important to find and fix than low severity ones. Thus, a UEM that detects a higher percentage of the high severity problems will have more utility than, say, a UEM that detects larger numbers of usability problems, but ones that are mostly low severity (even though all problems found might be "real" by the definition used). Weighting thoroughness with severity ratings provides a measure that would reveal a UEM's ability to find all problems at all severity levels. Such a measure can be defined by starting with the definition of thoroughness:

$$\textit{Thoroughness} = \frac{\text{number of real problems found}}{\text{number of real problems that exist}} \qquad (1)$$

and substituting weighted counts instead of simple counts of problem instances:

$$\textit{Weighted Thoroughness (by severity)} = \frac{\sum \text{s}\left(\text{rpf}_i\right)}{\sum \text{s}\left(\text{rpe}_i\right)} \qquad (14)$$

where s(u) is the severity of usability problem u, rpf_i is the ith real problem found by the UEM in the target system, and rpe_i is the ith real problem that exists in the target system. This kind of measure gives less credit to UEMs finding mostly low severity problems than ones finding mostly high severity problems.

However, for many practitioners who want UEMs to find high severity problems and not even be bothered by low severity problems, this kind of thoroughness measure does not go far enough in terms of cost effectiveness. For them, perhaps the breakdown of thoroughness at each level of severity is better:

$$Thoroughness(s) = \frac{\text{number of real problems found at severity level s}}{\text{number of real problems that exist at severity level s}} \tag{15}$$

Practitioners will be most interested in thoroughness for high levels of severity and can ignore thoroughness for low severity. Or a measure of the average severity of problems found by a given UEM, independent of thoroughness, might be more to the point for some practitioners:

$$s_{\text{avg}}\ (UEM_A) = \frac{\sum \mathrm{s}\ (\mathrm{rpf_i})}{\text{number of real problems found by UEM}_{\text{A}}} \tag{16}$$

The average severity found by a given UEM could be compared to the same measure for other UEMs or to the same measure for the problems existing in the target system:

$$s_{avg}\ (exist) = \frac{\sum s\ (rpe_i)}{\text{number of real problems that exist}} \tag{17}$$

The above definition would identify UEMs good at finding the most important problems, even ones that do not score the highest in overall thoroughness.

If researchers believe severity is important enough, they can include it in the ultimate and actual criteria as another way to enhance the criterion definition. By including severity, researchers introduce the problem of finding an effective actual criterion that captures "severity-ness," since no absolute way to determine the "real severity" of a given usability problem exists. Nonetheless, there are numerous schemes for subjectively determining severity ratings for usability problems. Nielsen (1994b) is a representative example. Rubin (1994) uses a criticality rating combining severity and probability of occurrence.

Frequency Data

Another extension can be made using data that accounts for the frequency of each usability problem in the real set of usability problems. Frequency counts are most useful for data from a lab-based usability test where it is clear the most important problems are those experienced by a higher percentage of users. Using the same approach discussed for severity ratings, thoroughness can be refined by substituting frequency counts instead of simple counts as shown in the following equation:

$$\textit{Weighted Thoroughness (by frequency)} = \frac{\sum f\left(rpf_i\right)}{\sum f\left(rpe_i\right)} \qquad (18)$$

where f(u) is the frequency of usability problem u, rpf_i is the ith real problem found by the inspection method in the target system, and rpe_i is the ith real problem that exists in the target system. Thus, weighted thoroughness (by frequency) provides a measure that reveals an inspection method's ability to find problems at all frequency levels.

Usability

Very little of the research involving UEM studies even mention UEM usability as an attribute or comparison measure worth investigating. Mack and Nielsen (1994) have clearly demonstrated that learnability and usability of inspection methods has been a frequent complaint of many researchers and practitioners. Some methods such as the cognitive walkthrough require extensive learning, going beyond the resources many development organizations have in terms of time and personnel. Therefore, usability is an additional measure that should be an important part of UEM criterion selection.

Review of UEM Studies

To determine if current UEMs add value to the HCI design and evaluation process, researchers design experimental studies to investigate various aspects of effectiveness. Often, UEM comparison studies use validity and thoroughness measures as core characteristics of effectiveness. In addition, researchers often include issues such as expertise requirements and time demands to use a particular method as a way to make a claim about the method. The goal of most experimental comparison studies is to establish sound conclusions regarding the effectiveness of a particular method when compared to other similar methods. In many scientific disciplines, researchers often use meta-analysis techniques to accumulate experimental and correlational results across independent studies. For a meta-analysis to be complete, studies focusing on a particular issue need to be in abundance. In addition, studies should provide the necessary descriptive statistics to calculate effect sizes of the important differences between methods. Because of the youth of HCI, UEM comparison studies are not abundant and many studies do not provide the necessary statistics to do an appropriate meta-analysis. However, enough data do exist to look at several important characteristics across popular studies.

Table 2-4 presents a summary of 19 studies representing some of the more popular UEM comparisons. The comparison is normally of one UEM to others or of one UEM as it relates to manipulation of a particular variable (e.g., expertise, software application). The 19 studies identified here do not represent the entire population of UEM comparison studies. A full analysis of UEM comparison studies might yield another 19 where the focus is expanded to issues such as teams vs. individuals, cost, guidelines vs. no guidelines, etc.

A majority of the UEM comparison studies (14) used the thoroughness measure for comparison. Examining the thoroughness studies in closer detail, 7 of the 14 studies specifically used the heuristic evaluation technique in comparison to other methods. Three studies (Nielsen, 1992; Nielsen & Molich, 1990; Virzi, 1992) did not compare other methods, but rather examined a particular method in terms of individual evaluator thoroughness. The heuristic evaluation technique is reported as having a higher thoroughness rating in 6 out of these 7 studies (85.7%). Thus, a natural conclusion from the thoroughness criterion is that the heuristic evaluation appears to find more problems than other UEMs when compared head-to-head. Such a conclusion is often reported in the literature with only a few exceptions.

The results from using the validity measure are not so clear; providing mixed conclusions in terms of a UEM that might be more effective for focusing on particular problems. A common way to examine validity is in terms of problems that impact users; often collected through lab-based testing with users. However, many of the studies in Table 2-4 do not necessarily identify if a lab-based usability test was used to determine the validity measure. One study (i.e., John & Marks, 1997) looked at validity in terms of the number of problems changed by the developer. When researchers focus on different aspects of validity, it is difficult to provide consistent

TABLE 2-4. Summary of UEM Effectiveness Studies (codes explained at end of table).

Study	Methods (subjects)	Thoroughness	Validity	Severity	Expertise	Time to Use Method	Notes
Bastien and Scapin (1995)	EC (10) NM (10)	EC > NM EC (M=89.9, SD = 26.2) NM (M=77.8, SD=20.7) p<.03					• NM = No method. Subjects just listed problems without a method guiding them. • Study provided mean, stddev, and p-values.
Bastien et al. (1996)	EC (6) ISO (5) NM (6)	EC > ISO/NM EC (M=86.2, SD=12.7) ISO (M=61.8, SD=15.8) NM (M=62.2, SD=13.8) p <.01					• Study provided mean, stddev, and p-values.
Beer et al. (1997)	CW (6) TA (6)	TA > CW p < .001		TA > CW p < .001			• TA > CW for major, minor, and cosmetic problems
Cuomo and Bowen (1992) Cuomo and Bowen (1994)	HE (2) CW (2) GR (1)	GR > HE > CW	CW > HE > GR CW (58%) HE (46%) GR (22%)			GR > CW > HE	• Not Reported: mean, stddev, and p-values. • CW: Team approach.
Desurvire et al. (1991)	HE				EX > U > NE		• Not Reported: n • Reported R^2 for predicting task completion rates.
Desurvire et al. (1992) Desurvire and Thomas (1993)	HE (3) CW (3) PAVE (3) UT (18)		HE > PAVE > CW HE (44%) PAVE (37%) CW (28%)	HE > CW	EX > DV > NE (Validity & Severity)		• Not Reported: stddev and p-values. • PAVE improved DV & NE performance.
Doubleday et al. (1997)	HE (5) UT (20)	HE > UT HE (86) UT (38)					• Not Reported: mean, stddev, and p-values. • 39% of UT problems not identified by HE. • 40% of HE problems not identified by UT.
Dutt and Johnson (1994)	HE (3) CW (3)	HE > CW		HE > CW			• Not Reported: percentage, mean, stddev, and p-values.

TABLE 2-4 (Continued)

Study	Methods (subjects)	Thoroughness	Validity	Severity	Expertise	Time to Use Method	Notes
Jeffries et al. (1991)	HE (4) CW (3) GR (3) UT (6)	HE > CW/GR > UT HE (50%) CW (17%) GR (17%) UT (16%)		UT > HE > GR > CW p <.01		UT > CW > GR > HE	• Not Reported: stddev. • HE also found highest number of least severe problems. • CW & GR essentially used a team of 3 people, not individuals.
John and Marks (1997)	CA (1) CW (1) GOMS (1) HE (1) UAN (1) SPEC (1)	HE > SPEC > GOMS > CW > CA > CA > UAN HE (31%) SPEC (24%) GOMS (16%) CW (15%) CA (.08%) UAN (.06%)	CW > SPEC > GOMS > HE > CA/UAN CW (73%) SPEC (39%) GOMS (30%) HE (17%) CA/UAN (0%)				• Not Reported: stddev and p-values. • Validity here is the number of problems changed by developer.
John and Mashyna (1997)	CW (1) UT (4)		CW (5%)				• Not Reported: mean, stddev, and p-values. • Case study approach.
Karat et al. (1992)	IW (6) TW (6) UT (6)	UT > TW > IW p <.01		UT > TW > IW p <.01 (for System 1)			• Not Reported: percentage and stddev. • Walkthroughs essentially used Heuristics for evaluation. • Evaluated 2 different systems, but did not characterize the difference between the 2 systems.
Nielsen and Molich (1990)	HE (various)	HE problems found: 20% to 51% (M)					• Not Reported: stddev and p-values. • Compared different systems using HE.
Nielsen (1990a)	TA (36)		TA found 49% (M) of problems				• Not Reported: stddev and p-values.
Nielsen (1992)	HE (31 NE, 19 EX, 14 DE)	HE overall average across 6 systems was 35%			DE > EX > NE DE (60%) EX (41%) NE (22%)		• Not Reported: stddev. • Nielsen collapsed 6 HE studies

TABLE 2-4 (Continued)

Study	Methods (subjects)	Thoroughness	Validity	Severity	Expertise	Time to Use Method	Notes
Sears (1997)	HE (6) CW (7) HW (7)	HE > HW > CW (combining 4 or 5 evaluators) HW > HE > CW (combining 2 or 3 evaluators)	HW > CW > HE	HW > HE > CW p <.01			• UT used to determine actual problems. • No mean or stddev reported for thoroughness or validity.
Virzi et al. (1993)	HE (6) TA (10) UT (10)	HE > TA > UT HE (81%) TA (69%) UT (46%)					• Not Reported: stddev and p-values.
Virzi (1990)	TA (20)		TA found 36% (M) of problems				• Not Reported: stddev and p-values.
Virzi (1992)	TA (12)	M=32%, SD=.14					• Reported overall detection rate for individuals.

Codes for Methods
HE = Heuristic Evaluation
CW = Cognitive Walkthrough
GR = Guidelines Review
UT = Usability Lab Test
HW = Heuristic Walkthrough
CA = Claims Analysis
NM = No method
UAN = User Action Notation
SPEC = Reading the Specification
IW = Individual Walkthrough (essentially used Heuristics, not CW process)
TW = Team Walkthrough (essentially used Heuristics, not CW process)
TA = Thinking Aloud
PAVE = Programmed Amplification of Valuable Experts
EC = Ergonomic Criteria
GOMS = Goals, Operators, Methods, & Selection Rules

Codes for Expertise
EX = Experts
DE = Double Experts
DV = Developers
NE = Non-experts
U = Users

comparisons. In addition, researchers do not always provide the necessary details to determine if standard methods were used in collecting the problems from user testing.

In addition to thoroughness and validity, many researchers also look at the severity of problems identified by a particular method. Severity information is often easy to collect during or after the evaluation, but there is no consistent definition of severity across studies. Some researchers use the severity definitions originally provided by Nielsen (1994b), while others use severity definitions developed as part of the study (e.g., Desurvire et al., 1992; Desurvire & Thomas, 1993; Dutt et al., 1994; Karat et al., 1992). Referencing Table 2-4, severity comparisons fall into the same situation as validity; that is, no single method consistently identifies the most severe problems. Although the heuristic evaluation technique is often criticized for identifying a large number of minor problems (e.g., Doubleday et al., 1997; Jeffries et al., 1991), such a conclusion is not fully supported from the six studies in Table 2-4 where researchers focused on severity ratings.

Another important consideration for researchers examining UEMs is the difference between experts and non-experts using a particular method. Evaluators classified as experts typically have experience in user interface design and evaluation. Although it might be beneficial in some HCI environments to have non-experts use methods equally as well as experts, the reality is that experts generally perform much better than non-experts (Desurvire et al., 1991; Desurvire et al., 1992; Jeffries & Desurvire, 1992). The three studies in Table 2-4 reporting on expertise comparisons support the conclusion that experts are much better at finding important problems than other evaluators such as developers, non-experts, and users. In fact, Nielsen (1992) found that double experts (i.e., experience in user interface design and the system application) were the most effective at finding problems using the heuristic evaluation technique.

Although not frequently reported, the time to use a particular method is a valuable measure to practitioners where resource efficiency is an issue. Only 2 of the 19 studies in Table 2-4 specifically reported on the time to use a particular method. The conclusion from these two studies is that the heuristic evaluation method requires the least amount of time when compared to methods such as the cognitive walkthrough, user testing, and guideline reviews. In fact, Jeffries et al. (1991) noted that the heuristic evaluation method was most effective when considering problems found per person-hour. The literature also seems to support the fact that the cognitive walkthrough is very time intensive, thus reducing its overall effectiveness (Lewis et al., 1990; Rowley & Rhoades, 1992; Wharton et al., 1992).

Summary

Although categories of UEMs are becoming somewhat well-defined in the HCI discipline, techniques for evaluating and comparing UEM effectiveness are not yet well-established. However, it is still possible to develop stable and consistent criteria for UEM effectiveness. Thoroughness, validity, and reliability appear to form the core of criterion measures researchers should continue to investigate. Thoroughness and validity measures must take into account the question of usability problem realness. Currently, lab-based testing with users appears to be an effective way to provide a "standard" set of real usability problems. Although not an exact replication of real work contexts, user-based lab testing does provide a good indication of the types of problems that actually impact users. Another possibility for researchers is to push for examining problems that real users do encounter in real world contexts using field studies and remote usability evaluation (Hartson & Castillo, 1998). The main difficulty with these methods, however, is that the lack of controls on tasks users perform can mean an inability to compare results among such UEMs.

In the near term, both usability researchers and usability practitioners will benefit from methods and tools designed to support UEMs by facilitating usability problem classification, analysis, reporting, and documentation, as well as usability problem data management (Hartson, Andre, Williges, & Van Rens, 1999). In the context of UEM evaluation, a reliable usability problem classification technique is essential for comparing usability problem descriptions, required at more than one point in UEM studies.

Finally, researchers should consider ways to reduce criterion deficiency and criterion contamination. One of the easiest ways to reduce criterion deficiency is through the use of several measures in the actual criterion, each focusing on a different characteristic of the UEM. In addition, examining how multiple measures can be combined into a composite measure that has a stronger relationship to the ultimate criteria is a possibility.

At this point in the HCI field, it appears to be nearly impossible to do an appropriate meta-comparison of usability studies. Two primary issues contribute to the challenge of comparing UEMs. First, UEMs are extremely young (less than ten years) when compared to social science disciplines where baseline studies are frequently performed. Because of its youth, the number of baseline comparative studies is almost non-existent. Second, the methods for usability evaluation are not stable. In fact, they continue to change because human-computer systems, their interaction components, and their evaluation needs change rapidly, requiring new kinds of UEMs and constant improvement and modifications to existing UEMs.

CHAPTER 3. DEVELOPMENT OF THE USABILITY PROBLEM INSPECTOR

Over the past few years, students and faculty in the Usability Methods Research Laboratory at Virginia Tech have been developing several different methods and tools to support the usability practitioner beyond the usual formative usability evaluation in an interaction development process for ensuring usability. The earliest work began with the Usability Problem Taxonomy (Keenan, 1996; Keenan, Hartson, Kafura, & Schulman, 1999), postulated on the view that each usability problem possessed attributes in both a task- and an artifact-related dimension (Carroll, Kellogg, & Rosson, 1991). The artifact dimension contained three major categories (Visualness, Language, and Manipulation), while the task dimension contained two major categories (Task-Mapping and Task-Facilitation). The resulting taxonomy consisted of four levels of problem types and one level of specific examples of usability problems. Summative evaluation indicated that the Usability Problem Taxonomy yielded acceptable reliability at the first classification level on the artifact dimension ($\kappa = .403$), but only marginal reliability on the task dimension ($\kappa = .095$). Results indicated that categories at lower levels of classification were less clear. Subjects frequently complained of obscure wording, non-intuitive category names, and non-distinct categories at these levels.

Building on the Usability Problem Taxonomy, van Rens (1997) expanded the work begun by Keenan (1996) by creating the Usability Problem Classifier, with a new structure and adding much new content. The first version of the Usability Problem Classifier departed from its predecessor by shifting primary focus to the object type. Classification began with identification of the type of object (e.g., menu, button) most closely associated with the usability problem being classified. Although some improvement occurred, many of the shortcomings identified within the Usability Problem Taxonomy persisted in the first version of the Usability Problem Classifier. After an extensive iterative design process, van Rens determined the primary focus on object type unnecessarily complicated classification efforts. Consequently, van Rens concluded an entirely new starting point would be required to converge on a better design.

In a second version of the Usability Problem Classifier, van Rens (1997) integrated the previously separate task and object classification paths into a single sequence. Within the resulting hierarchy, categories were ordered in decreasing order of severity, potentially preventing problem under-rating.

Formative evaluation confirmed that the second version of the Usability Problem Classifier resolved many of the issues present in the earlier versions. However, more comprehensive evaluations brought additional issues, including redundancy at the lower nodes of the taxonomy, verbose and unwieldy wording, and gaps in the taxonomy. Andre, Belz, McCreary, and Hartson (in press) developed a third version of the Usability Problem Classifier to address these issues. The most significant change in this third version required users to identify *when a usability problem occurred* (i.e., before, during, or after) prior to beginning classification of the *type of problem*. Previously, this decision branch was found towards the bottom of the classification hierarchy. Locating this decision branch earlier in the classification hierarchy facilitated more meaningful classification of task attributes. Figure 3-1 shows the third version of the Usability Problem Classifier with the before, during, and after decision located first in the hierarchy followed by task attributes and finally the object components relevant for each category.

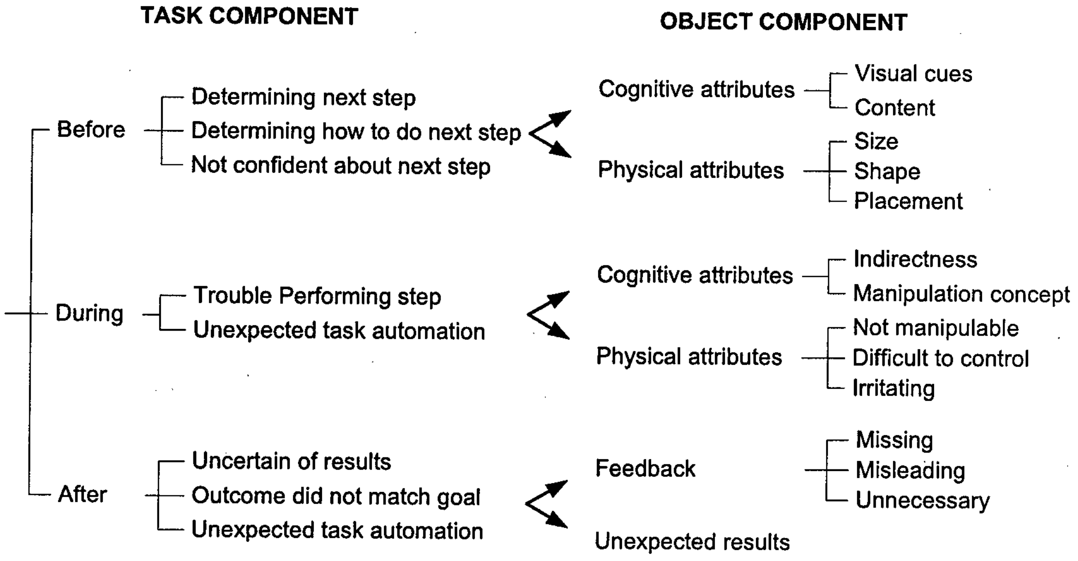


FIGURE 3-1. The Usability Problem Classifier with Before, During, and After Decision Nodes.

THEORY-BASED INTEGRATING MODEL: THE USER ACTION FRAMEWORK

Classifying problems based on when a usability problem occurred provided a simple approach to describing problems. However, when using such a strategy, evaluators did not always understand how to classify a usability problem as occurring before, during, or after the associated user action. Evaluators needed a framework to classify problems based on user interaction activities.

The solution came from Norman's (1986) theory of action. Norman's model describes a user's interaction with the computer as occurring in seven stages:

1. Establishing the goal.
2. Forming the intention.
3. Specifying the action sequence.
4. Executing the action.
5. Perceiving the system state.
6. Interpreting the state.
7. Evaluating the system state with respect to the goals and intentions.

The "before user action" corresponded approximately to the first three stages of Norman's (1986) model, the "after action" part corresponded roughly to the last three stages, and the "during action" part was something of a match to Norman's fourth stage: executing the action. Adapting and extending Norman's model helped form the Interaction Cycle shown in Figure 3-2. Instead of using a decision node based on when a usability problem occurred (i.e., before, during, or after), the Interaction Cycle uses a primary decision node based on user activities in interaction-based systems: Planning, Physical Actions, and Assessment. The Interaction Cycle is then used as the top-level organizing structure for usability concepts and issues obtained from the Usability Problem Classifier to form the User Action Framework (UAF), comprised of the Interaction Cycle, plus the structured knowledge base of usability issues and concepts, as shown in the whole of Figure 3-3. Appendix A provides a detailed diagram of the UAF, showing the Interaction Cycle parts at the top level and the hierarchically structured knowledge base at deeper levels within the framework.

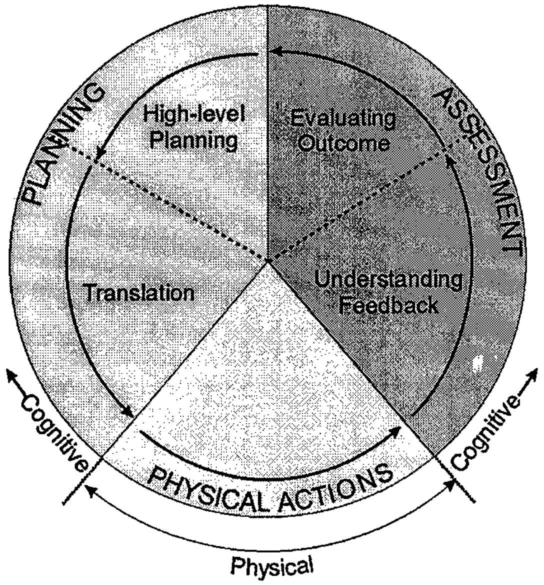


FIGURE 3-2. The Interaction Cycle.

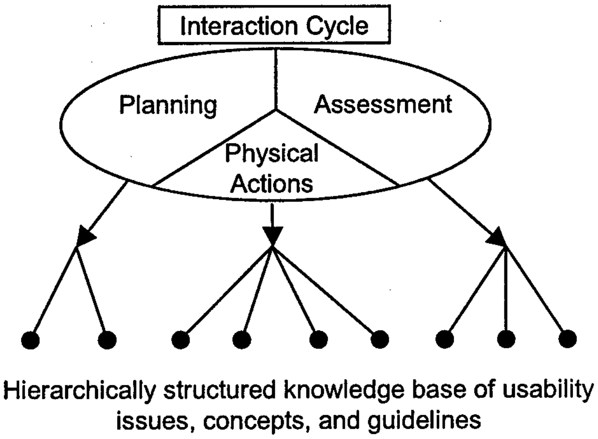

Interaction Cycle
Planning
Assessment
Physical Actions
Hierarchically structured knowledge base of usability issues, concepts, and guidelines

FIGURE 3-3. Forming the UAF from the Interaction Cycle and Structured Knowledge Base.

Description of the Interaction Cycle

The Interaction Cycle organizes the underlying knowledge base of usability concepts and issues of the UAF within a cycle of cognitive and physical user actions involved in the performance of a task using a computer. Classification of a usability situation within the usability concept space begins by associating it with the appropriate part in the Interaction Cycle. These associations are based on usability attributes that have salient characteristics represented in each Interaction Cycle part.

In addition to user activities represented in the Interaction Cycle, certain system activities interact with user physical actions in a complimentary fashion as shown in Figure 3-4. The System Interaction Cycle, shown on the right side of Figure 3-4, helps to complete a cycle that involves user input, system processing and representation of outcome, and user evaluation of the outcome. The User Interaction Cycle, herein referred to as the Interaction Cycle, organizes the majority of usability concepts and issues first developed in the Usability Problem Classifier, although some problems can be uniquely associated with the System Interaction Cycle. Both cycles help to complete the unique structure and content that defines the UAF.

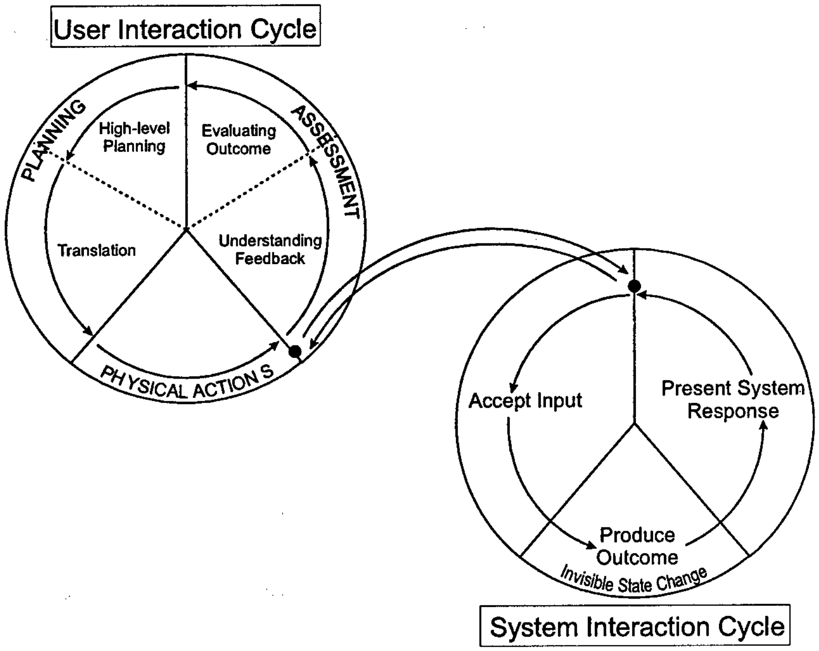


FIGURE 3-4. User Interaction Cycle Combined with a System Interaction Cycle.

Structure and Content of the UAF

Planning

Planning occurs when users determine what actions need to be taken and how to go about completing those actions. Planning includes all thinking by users to work out what to do and how to do it. Plans are established in the form of a hierarchy of plan entities detailed in Table 3-1. Planning breaks down into High-Level Planning and Translation, each of which is an important part of the Interaction Cycle in its own right.

TABLE 3-1. Hierarchy of Plan Entities.

Planning level	**Domain**	**Example**	**Description of level**
Goal	Always work/problem domain	Produce business letter	What is to be achieved in work domain
Task	Planning tasks to be done using computer	Formatting the page	Decomposition of work goals into computer-based tasks
Intention	Planning intentions to be done using computer	User intends to set left margin	Always the lowest level plan, just above physical actions
Action plan	Plan for physical actions to be done on computer	User plans to click and drag an icon	Only level where plan is for physical actions on interface objects

High-Level Planning

High-level planning is concerned with the user's ability to understand the overall computer application in the perspective of work context, problem domain, environmental requirements and constraints. The primary focus is on the system model and metaphors, and the user's knowledge of system state and modalities. High-level planning includes user work goal decomposition across a hierarchy of plan entities: goals, task, and intentions, all expressed in cognitive problem-domain language.

Translation

The user must translate intentions into plans for physical action(s). Translation is largely about cognitive affordances to support the users' ability to plan physical actions. The user draws on knowledge, experience, and cognitive affordances in the interaction design to determine, establish, or ascertain an action plan to carry out the intention. The activity here involves translating from the language of the problem domain to the language of actions upon user interface objects. For example, the intention to make a word bold in some text might result in a plan to drag-select the word and click on the "bold" button.

The Translation sub-part is perhaps the most difficult for both designer and user, and accounts for a large proportion of usability problems observed in the field. Translation is purely cognitive: the user has formed a mental plan for actions, but has not yet done those actions. Usability issues in the UAF under Translation include those that pertain to presentation of the cognitive affordances (e.g., perceptual issues, legibility, noticeability, timing, layout, complexity, consistency, presentation medium, and supporting human memory limits). Translation issues also include effectiveness of content or cognitive affordance meaning (e.g., issues of clarity, completeness, error avoidance, consistency, and the ability to predict the effects of a given user action). Translation also includes preferences and efficiency issues (e.g., user control, task structure, number of steps, and short cuts).

Physical Action

Executing the planned actions is the focus of the Physical Action part of the Interaction Cycle. The Physical Action part of the Interaction Cycle is about perceiving objects to manipulate and manipulating objects. Perception has to do with the usual factors of noticeability, legibility, contrast, and timing. Object manipulation has to do with interaction complexity, input/output devices, interaction styles and techniques, manual dexterity, layout (Fitts' law), and physical disabilities.

Outcome (System Interaction Cycle)

As a result of physical user actions, the system computes an outcome, an internal and invisible state change. The system then produces a system response, providing feedback representing the outcome to the user. The only way a user can know something about the outcome is indirectly by way of the system response. Therefore, the outcome is primarily a System Interaction Cycle issue, but can have usability consequences. Outcome issues relating to usability problems include system automation, locus of control, and system errors.

Assessment

The Assessment part parallels the Translation part in that it has to do with presentation of feedback, meaning of feedback, and preferences and efficiency. However, these issues now apply to the feedback and how it supports the user's ability to gauge the outcome of physical actions. In addition to the usual issues of legibility, noticeability, timing, layout, grouping, presentation of feedback also involves complexity, clutter, consistency, organization of information displays, and presentation medium. Effectiveness of feedback content and meaning, as one might guess, depends on clarity, completeness, sufficiency, correctness, relevance, and consistency of feedback.

An early version of the UAF also included an *independent* part to the Interaction Cycle. The independent part captured issues that were not necessarily task-based problems. These independent issues did not seem to fit within the Planning, Physical Action, and Assessment parts of the Interaction Cycle. Thus, the independent part allowed for placement of problems that were often about overall design issues not related to a particular part of the Interaction Cycle. Initial testing of the Interaction Cycle with an independent part did not add significant value to the UAF structure (reference Chapter 4).

Moving Through the Parts of the Interaction Cycle

Figure 3-5 shows in more detail the Interaction Cycle parts, corresponding to cognitive and physical actions users make while performing a task using a computer. Most user-system tasks typically start with Planning and move sequentially through the rest of the Interaction Cycle. However, various inputs can start the process at other points in the Interaction Cycle.

Sequential Flow

Consider a user working on the goal of composing a business letter. A new task arises in the Planning part of the Interaction Cycle, the task to print the document. In the first intention (i.e., the "getting started intention") the user intends to invoke the print function. The user might not do more planning at this point, expecting the outcome of this intention to lead to the next natural intention. The user translates this first intention to an action plan by deciding to select Print from the File menu, using experiential knowledge or cognitive affordance provided by the menu. The user then executes the physical action by selecting the Print menu choice. The system accepts the menu choice, computes, and displays a Print dialogue box as a new state. The user perceives the dialogue box and interprets/understands it. Continu:ng with Assessment, the user decides that the dialogue box makes sense at this point in the interaction; thus, the task is on track. Having made a turn around the Interaction Cycle, the user returns to the Planning portion to formulate the next intention to deal with the dialogue box.

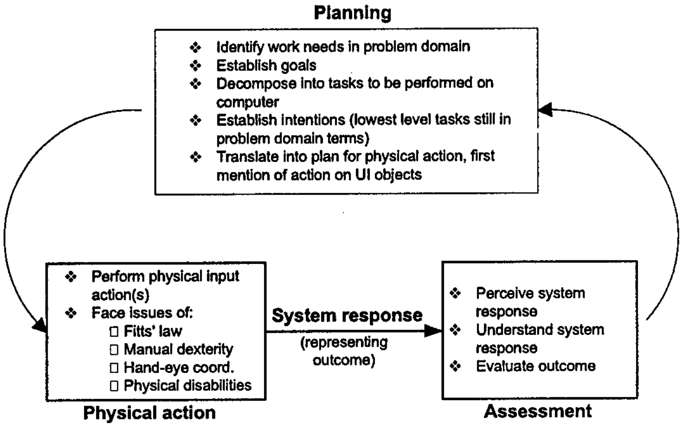

Planning
❖ Identify work needs in problem domain
❖ Establish goals
❖ Decompose into tasks to be performed on computer
❖ Establish intentions (lowest level tasks still in problem domain terms)
❖ Translate into plan for physical action, first mention of action on UI objects
❖ Perform physical input action(s)
❖ Face issues of:
□ Fitts' law
□ Manual dexterity
□ Hand-eye coord.
□ Physical disabilities
Physical action
System response
(representing outcome)
❖ Perceive system response
❖ Understand system response
❖ Evaluate outcome
Assessment

FIGURE 3-5. Representation of Process Flow Through Interaction Cycle Parts.

Variations in Flow

The flow of interaction is not limited to a sequential path through the cycle. Different kinds of users and usage situations lead to different paths through the cycle and different support needs in interaction design. Some factors affecting flow of interaction within the cycle include:

- <u>Level of task focus</u>. Depending on expertise and recent practice, users think of tasks and intentions at higher or lower levels of abstraction, affecting how many times they traverse the Interaction Cycle to accomplish the same thing. For example, expert users will automate many actions, without conscious thought of individual intentions or translation to explicit action descriptions.

- <u>Intention shifts</u>. Sometimes a user changes intentions, tasks, or goals during interaction. A user who decides, in the Assessment part of the cycle, that an error has occurred will usually shift to an error recovery task. Intention shifts require cognitive stacking involving usability issues about human memory limits.

- <u>Input/output overlap</u>. In fine-grained view of a direct manipulation operation involving hand-eye coordination (such as dragging), user input and perception of output overlap. This kind of tightly coupled interaction, referred to by (Draper, 1986) as inter-referential input/output, also requires overlap of system input and output processing.

- Cycles that begin with an output display. The usual cycle begins with Planning, Physical Action, etc. Some excursions around the Interaction Cycle, however, begin with an output display. A real time process control application, for example, can respond to stimuli from the outside environment and autonomously display a message to draw attention to an out-of-limits parameter. Sutcliffe et al. (1996) call this output display starting point a system initiative task. For the user, this cycle begins with perception of the display and proceeds around with Assessment and Planning to formulate a task and intention to deal with the situation. A similar situation occurs when, in a cooperative work environment, the interaction cycles of two or more users interleave so that one user is perceiving and interpreting the output from another user's input actions.

Norman (1986) emphasizes that real activity does not progress as a simple sequence of stages. Stages appear out of order, some may be skipped, some repeated. In some human-machine systems, the person is reactive (i.e., event or data driven), responding to events, as opposed to starting with goals and intentions. Consider the task of an operator in the control room of a nuclear power plant. The operator's task is to monitor the system state, continuously checking indicators for proper operation. When an indicator starts to move a bit out of range, or when something goes wrong and an alarm is triggered, the operator must diagnose the situation and respond appropriately. In this situation, evaluation leads to the formation of goals and intentions. Although this appears to make sense, an argument could be made that this situation originally begins with the operator goal of monitoring the system state, leading to an elaborate specification and execution of a visual cross-check of all relevant displays in the control room. The operator continues to repeat this cycle until an out-of-range indicator triggers the new formation of a goals and intentions. In either case, it seems reasonable to assume that user activity goes in a counter-clockwise direction around the Interaction Cycle, and even though user activity may start at, say, Assessment, the relationship between execution and evaluation is easily seen in human-computer systems.

Some researchers may argue that by developing a structure with hundreds of different problem types is equivalent to the enormous guideline approach such as the one Smith and Mosier (1986) developed. The difference is this structure is not just an "indented" structure (e.g., Kurosu et al., 1997), but rather a theory-based structure that parallels the way a user interacts with a system. Even though certain heuristics and design principles are covered throughout the Interaction Cycle, each path to the problem classification is different and aids in description.

The Importance of Reliability in the User Action Framework

Developing a reliable method was an important goal in designing the UAF. Users of usability evaluation tools have been known to experience significant variation when applying the various techniques. Without some level of reliability, one evaluator using a usability tool can get one result and another a different result, and the usability data for the project will depend on the individual using the tools. A good example is the popular heuristic evaluation technique developed by Nielsen and Molich (1990) in the early 90's. Heuristics, intended as a cheap, fast, and easy to use method for inspection of user interfaces, do not provide a structured framework to separate out the fine differences between various usability problems (Dutt et al., 1994).

In developing the UAF, designers recognized that reliable tool usage would depend on a consistent shared understanding of the underlying framework and how it is applied in specific cases. In particular, reliability of the UAF is interpreted in terms of consistency of classification across users. Formative evaluation observations made over several years during attempts to refine the UAF have led to an inescapable conclusion. Variation in interpretation of usability terms and concepts will occur. Designers will find it impossible to design a classification scheme to avoid this kind of variation. Therefore, a more realistic approach is to accommodate the variation inherent with any organized framework.

In early work with the UAF, the goal for classification reliability was for every user to take exactly the same classification path for a given usability situation (e.g., usability problem) being classified. When an evaluator subject failed to classify a problem on the expected path, the rationale for the errant classification was determined by interviewing the subject. Changes were made in the wording at each node where the user's classification path deviated from the expected choice. Semantic attractors were added to direct users to the likely issue while semantic deflectors were added at other nodes to direct users back to the likely classification path.

While the attractors and deflectors gave an initial boost in reliability, they failed as a sole strategy for steering all users into agreement on classification paths. A limit was soon reached where additional changes to avoid divergence by one user would work against previous changes made to avoid divergence in other users. Thus, designers found it impossible to converge on a single overall set of deflectors and attractors that would work the same for all users.

Subsequently, designers reasoned that classification reliability required only consistent final classification results, not identical classification paths. As a result, the design of the UAF deviated from a purely hierarchical structure, and instead, provided alternative paths for some classification choices. When classification paths for the same usability situation occasionally diverged, the UAF allowed for reconvergence on the same final classification node. This reconvergence was most effective where two usability attributes were more or less orthogonal. Consistency within a pure hierarchy forced the UAF users to always put the same attribute first in the classification sequence when, in fact, no such natural ordering existed, which meant adding artificial rules about which attributes to consider first and so on. A good example involves the usability attribute "preferences and efficiency." Preferences and efficiency attributes generally apply equally well to many other usability concepts. As an example, consider the situation illustrated in Figure 3-6. In this example, an error message might have attributes that describe its presentation or appearance and other attributes that describe its meaning or content. If a usability situation being classified involves preferences and efficiency issues about the presentation of an error message, then some users might choose feedback presentation first and preferences and efficiency second (top of Figure 3-6). Others might choose preferences and efficiency followed by feedback appearance (bottom of Figure 3-6). In either case, the user eventually selects an attribute relating to the format of the message and neither path is more correct than the other in arriving at this end point. By using alternate classification paths that both lead to the same combination of attributes, the UAF eliminates an artificial source of inconsistency that was present in the original design of the framework.

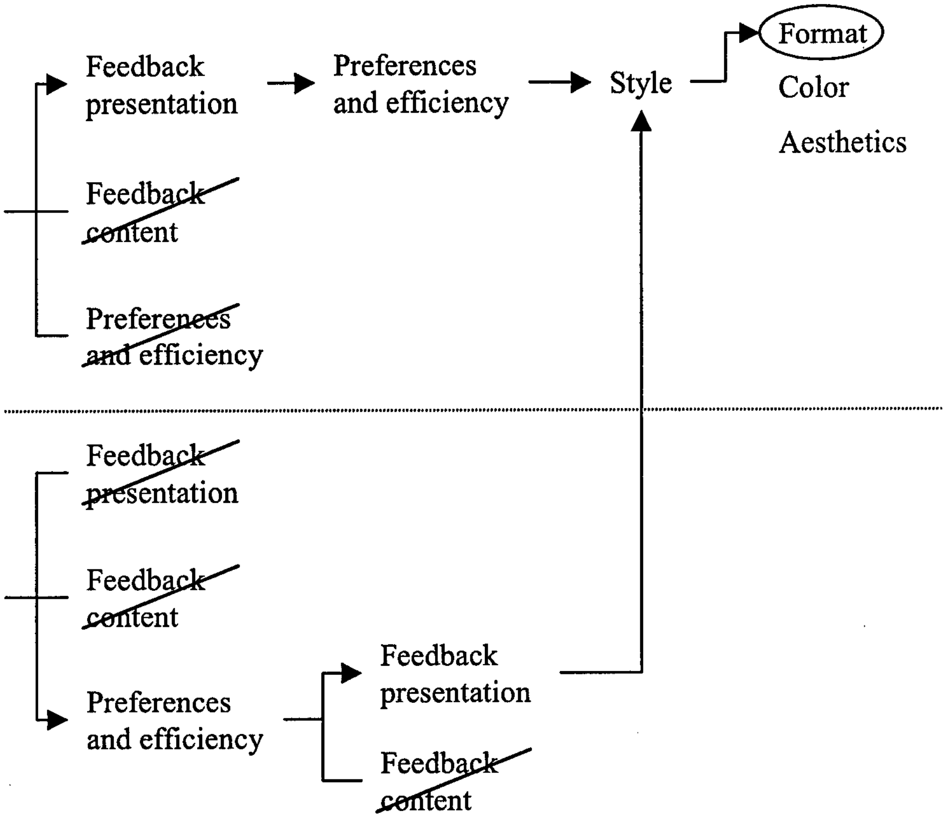
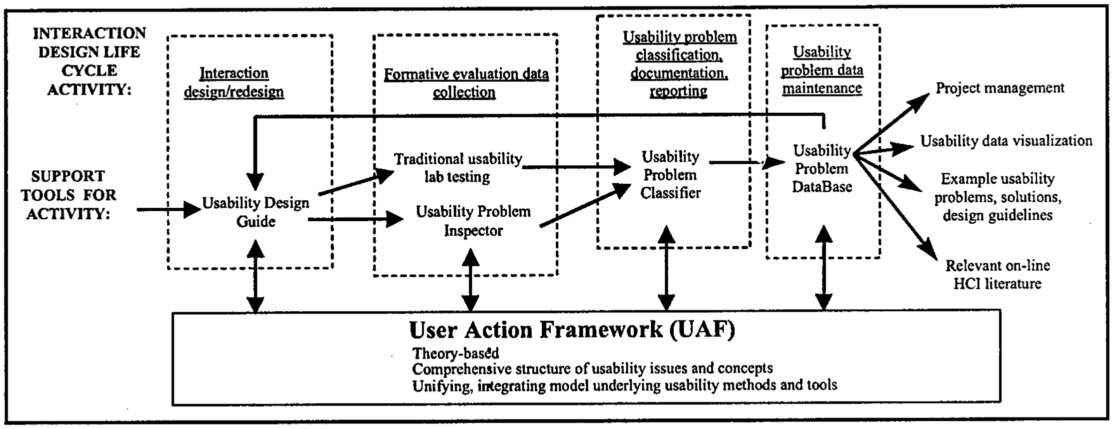


FIGURE 3-6. Alternative Paths to Classify a Usability Problem.

Integrating Usability Support Tools Within the UAF

During formative development of the UAF, research work expanded to include other usability engineering support methods and tools (e.g., usability inspection and usability data maintenance tools). Each tool required a structured way to organize usability concepts and issues in the context of the purpose of that tool. Rather than develop a structured organization separately for each new support tool, the UAF provided a unifying model by sharing the same consistent structure and content across all tools. In addition to linking usability support tools, the UAF provided a means for guiding usability development activities. Figure 3-7 illustrates how each usability development activity is supported by a tool and how integration of the tools via the UAF allows usability information to flow easily from tool to tool. The tools are integrated by sharing UAF content and structure as a common underlying framework. Each tool uses the content and structure of the UAF and locates a given usability situation in the same place within the UAF structure. As shown in Figure 3-7, four tools are currently integrated within the UAF:

- A Usability Design Guide to help initially develop interface design applications,
- A Usability Problem Inspector to help evaluators find problems once an interface design is ready for formative evaluation,
- A Usability Problem Classifier to help identify and label problems observed from lab-based testing or expert-based inspections, and
- A Usability Problem Database to support usability data maintenance within a project life cycle, and to support sharing and reuse of usability analysis and usability problem information and solutions that have worked in other similar situations.



INTERACTION DESIGN LIFE CYCLE ACTIVITY:
Interaction design/redesign
Formative evaluation data collection
Usability problem classification, documentation, reporting
Usability problem data maintenance
SUPPORT TOOLS FOR ACTIVITY:
Usability Design Guide
Traditional usability lab testing
Usability Problem Inspector
Usability Problem Classifier
Usability Problem DataBase
Project management
Usability data visualization
Example usability problems, solutions, design guidelines
Relevant on-line HCI literature
User Action Framework (UAF)
Theory-based
Comprehensive structure of usability issues and concepts
Unifying, integrating model underlying usability methods and tools

FIGURE 3-7. Broad Scope of Usability Tool Integration Provided by the UAF.

Developing the tools based on the UAF involves a mapping process. A mapping to a given tool retains the content and structure of the UAF, but changes the way the content is expressed; each knowledge item is rephrased into an *expression* reflecting the *purpose* of the tool. For example, consider this tool-neutral UAF expression of a usability concept in the Translation part of the UAF: "Request user confirmation to avoid errors with potentially destructive user action." When mapped to the Usability Problem Classifier tool, this concept is expressed as a classification-oriented question: "Is the *observed usability problem* about a potentially destructive error that might have been avoided by a user confirmation request?" The same concept maps to a different question in the Translation part of the Usability Problem Inspector tool: "At this point in task X being performed by a user in user class U, where a user action is potentially destructive, does the interaction design support error avoidance by requesting user confirmation?"

MAPPING TO THE USABILITY PROBLEM INSPECTOR

Overview

The mapping of the UAF to the Usability Problem Inspector (UPI) is currently the most mature tool and the subject of the research addressed in this work. The goal of the UPI is to help inspectors conduct a highly focused inspection of a target application resulting in a list of usability problems that users will potentially have with the application.

Tradeoffs

The design and use of the UPI involves tradeoffs between cost and performance. Cost includes issues such as complexity, training requirements, and time to apply the method. Performance issues include validity, thoroughness, and reliability of the method. The goal is to maximize performance parameters without a significant increase in cost. Although cost is a concern, the UPI is not intended to be a discount usability engineering method.

Usability Problem Inspector Users

Users of the UPI are *inspectors*, otherwise known as *evaluators*. These inspectors are usability practitioners who have: experience conducting formative and summative usability testing, formal training in HCI theory and models, and practical knowledge of HCI design principles. The UPI method is an *expert-based* usability inspection method, distinguishing it from a *user-based* method, such as the traditional kind of usability evaluation performed in the usability laboratory with users as participants. In a traditional lab-based usability test, expert evaluators conduct the evaluation, while users represent the vehicle for encountering problems with the application. In the UPI, the evaluator plays both roles by conducting the inspection and also representing the vehicle for encountering problems that users will potentially have in the application.

How the UPI Works

The UAF, when used in the context of the UPI brings together aspects of both the heuristic evaluation and cognitive walkthrough. While the heuristic evaluation focuses on ease of use, the cognitive walkthrough focuses on completeness and structure. The UPI fits in between these two, capturing the ease of use but also providing interaction-based structure. Figure 3-8 illustrates the overall process of using the UPI as it is mapped from the UAF to a specific inspection process. The following sections describe each of the components in Figure 3-8.

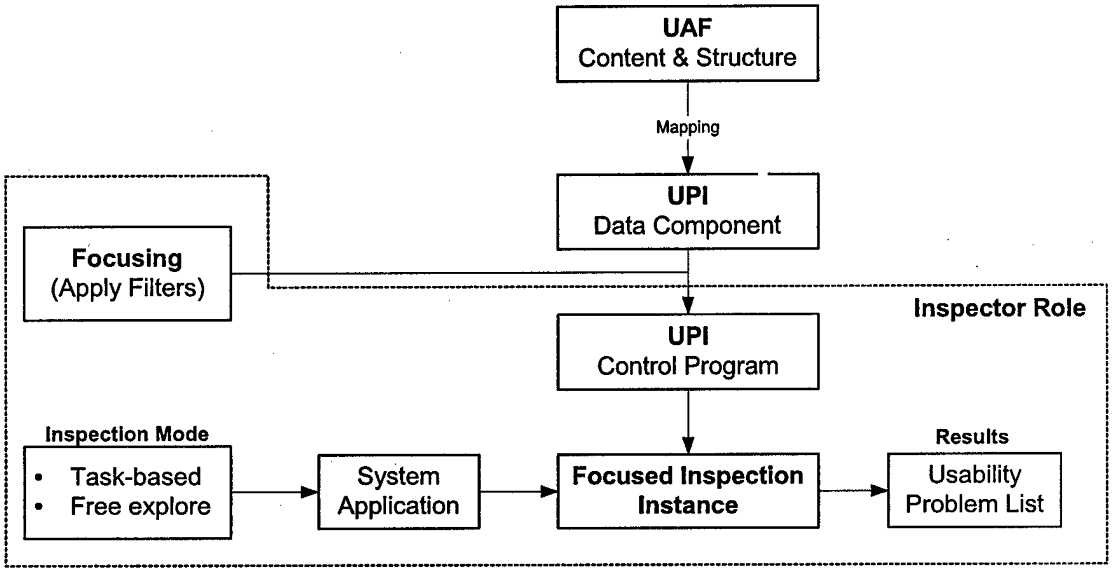


FIGURE 3-8. Process for Generating a Highly Focused Inspection Using the UPI.

UAF Content and Structure

The structured content of the UAF drives the inspection process, following interaction activities of user and systems much like that described by Norman (1986). Inspectors use the Interaction Cycle as a systematic entry framework to discover the potential problems in the user interaction design.

UPI Data Component

The UPI data component includes all the questions residing in the UPI and ready for use in an inspection process. As an example, consider one component in the Assessment part of the

Interaction Cycle: *perceptual issues*. Mapping to a UPI data component for this one issue results in the following:

UAF Content	UPI Data Component
Perceptual Issues	Is feedback presented so that user will easily notice?

Focusing

Evaluators of user interaction design often need to focus the set of questions used when inspecting a system. Such focusing helps to reduce the number of usability concepts and issues into a manageable set specific to the application. Adaptation of the inspection process to specific conditions and goals is accomplished by applying usability situation filters at the beginning of the inspection process and during structure traversal in the UPI. The user of the UPI provides two inputs for the filter when beginning the inspection process: target user class and interaction cycle focus.

Target User Classes

In usability inspection an expert performs usability evaluation "on behalf of" users. When performed on behalf of novice users, the inspection process needs to emphasize parts of the Interaction Cycle that present the most significant usability issues to novices (e.g., Planning/Translation). When experienced users are the target population, emphasis changes significantly within the Interaction Cycle – less emphasis on Planning/Translation and more on efficiency of Physical Actions. The inspector applies a usability situation filter for each task when using the UPI. In some cases, one user class filter may be used for all tasks because the users are novices at every aspect. In reality, users may be more experienced at certain tasks requiring the inspector to consider each task for possible filtering.

Interaction Cycle Focus

The inspector can also apply a filter to emphasize specific parts of the Interaction Cycle due to the nature of the software application. For example, the inspector may want to focus on error messages because field observations have shown that poorly presented error messages are "show-stoppers" for a particular software application domain. With a filter for error messages, the inspector mostly considers questions in the Assessment part of the Interaction Cycle, saving time and money by not performing an entire inspection.

UPI Control Program

The UPI control program implements the focusing provided by the inspector. The control program helps interpret focusing attributes associated with each data component in the UPI. For example, when the inspector applies a filter for an expert user class, the UPI control program looks for all UPI data components that have the appropriate value for "expert." All data components not associated with an expert user class are dropped from the possible questions the inspector will answer, thus reducing the question space.

Inspection Mode

To generate a focused inspection, the inspector uses the UPI in one of two modes: task-based or free-exploration. Representative tasks are generally developed and provided by the development team so that the inspector can "step through" important aspects of the design in a timely and efficient manner. In addition, the inspector can also use free-exploration as a way to further investigate specific attributes of the interface that were only briefly examined during the task-driven approach. For example, the inspector may have seen a dialog box during the task-driven approach that warrants further investigation. For the task part, the dialog box may not have generated usability problems but the investigator wants to look at some of the specific attributes of the dialog box, not related to any specific task other than user exploration.

System Application

The inspector focuses the inspection method through a particular inspection mode suitable to the target application under examination. Currently, the UPI is intended for graphical user interfaces, but is easily mapped to other interface styles such as the Web, voice response, and virtual environments.

Focused Inspection Instance

Through the process of focusing and using a particular inspection mode on a target system application, the inspector is able to conduct a highly focused inspection using the UPI. This highly focused inspection is unique to the individual inspector and represents the many goals and attributes of the system under examination. The inspector's time is used efficiently to only look at usability issues that result from selective focusing and using a particular inspection mode.

Results

The highly focused inspection results in a list of usability problems that users will potentially have with the system.

Implementing the UPI Tool

The first and current version of the UPI used HTML and Active Server Pages. Active Server Pages provide a means for the inspector to examine issues residing in the UAF database based on initial setup and answers to questions at each node in the framework. Users of the UPI are first presented with an Inspection Session Setup screen shown in Figure 3-9. Users select the relevant task and filter during the Inspection Session Setup. Task identification reminds the inspector of the current task under investigation and provides complete problem context information when the inspector eventually notes a usability problem. After clicking the CONTINUE button, inspectors are presented with the Inspection Session screen shown in Figure 3-10. In the Inspection Session screen, inspectors can view the current task and filter selected, review problem reports, and investigate the current problem statement shown near the bottom of Figure 3-10. Inspectors examine the current problem statement and decide if a potential issue exists in the context of the current task. By selecting YES in response to the problem statement, inspectors are presented with the next child node in the UAF database, thus traversing the depth of the structure. Selecting NO to a problem statement leads to the next sibling node in the UAF database, or to the next parent node if no sibling nodes exist at that point in the structure. Thus, by selecting NO, inspectors traverse the breadth of the UAF database structure until noting an issue. Inspectors are presented with a problem report form (Figure 3-11) by traversing the depth of a particular area in the UAF database until they reach an end node where specific usability attributes are listed for their selection. As shown in Figure 3-11, inspectors select one or more usability attributes relevant to the problem and provide a problem name and narrative description to complete the report form. The problem report form also records relevant information such as the current task, current filter applied, and inspection path taken by user to reach the end node (shown in the top portion of Figure 3-11). In the example shown for Figure 3-11, the inspector investigated Task 1, applied an Expert filter, and traversed the following path on the way to the displayed problem report form:

*Home \ Physical Actions \ Manipulating Affordances \ Physical Control *

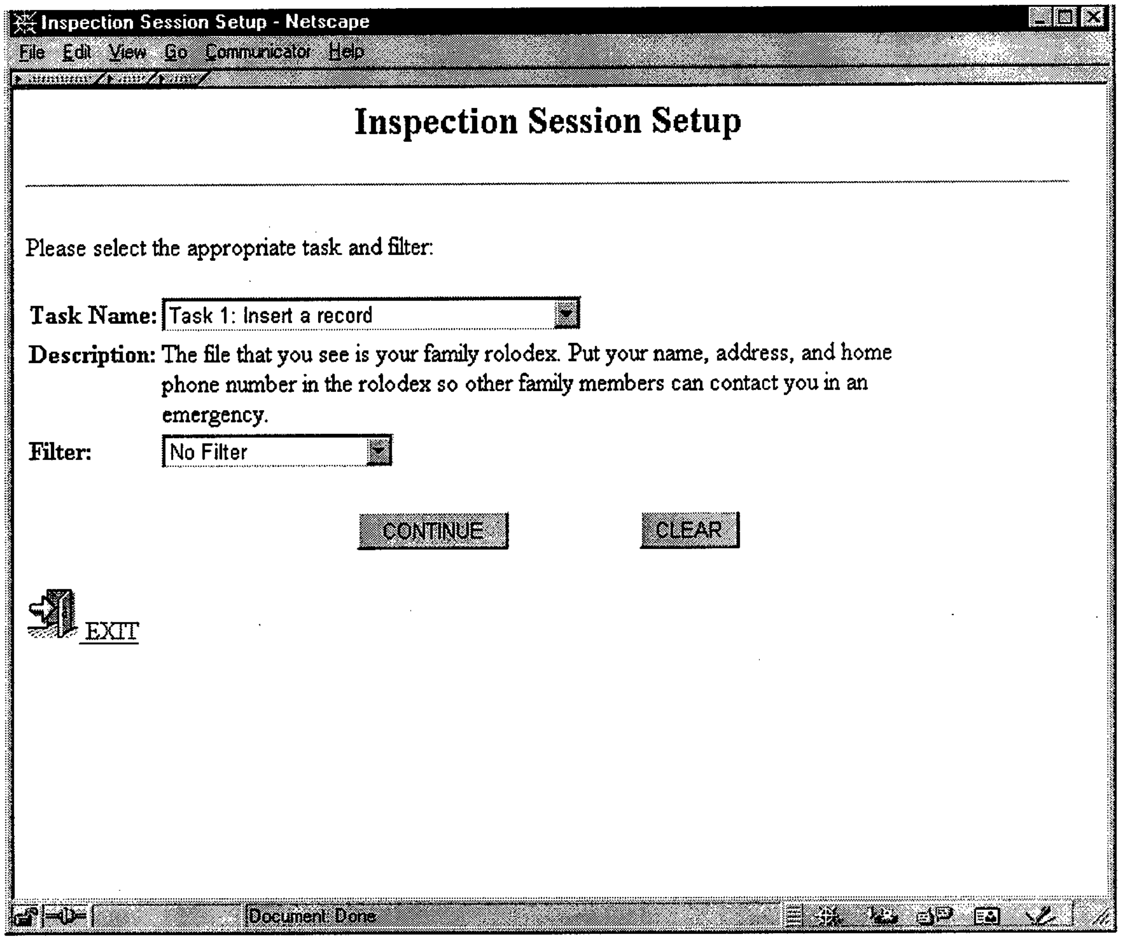
Once inspectors document a problem, they continue traversing the remaining structure of the UAF database, examining potential usability issues related to the current task. The inspector traverses the entire UAF database by selecting YES or NO to each usability problem statement until reaching the end of the framework where they are again presented with the Inspection Session Setup for the next task. Inspectors can also use the UPI in a free-exploration mode to investigate an interface design. The process during free-exploration mode remains the same as with the task-based approach; the only difference is that task information is not recorded. Free-exploration mode allows the inspector to report on potential usability issues without a specific task in mind. In addition, free-exploration mode allows the inspector to review in more detail interface areas that were only briefly encountered on the way to completing a task.

Differences from Other Methods

The UPI offers several important advantages over other expert-based inspection methods such as the cognitive walkthrough and heuristic evaluation.

Combines Ease of Use and Interaction-Based Structure

As an inspection tool, the UPI brings together aspects of both the heuristic evaluation and cognitive walkthrough. The UPI intends to capture the ease of use from the heuristic evaluation while also providing interaction-based structure as in the cognitive walkthrough. Because the UPI is based on a model of the way users interact with a system, it is intended to be an easier way to find and understand problems just as heuristics are intended to be easier than remembering hundreds of specific design principles. However, unlike the heuristic evaluation, the UPI provides more specific explanations of the problems because of the organized structure of usability concepts and issues found in the UAF.



Inspection Session Setup - Netscape
File Edit View Go Communicator Help
Document: Done

Inspection Session Setup

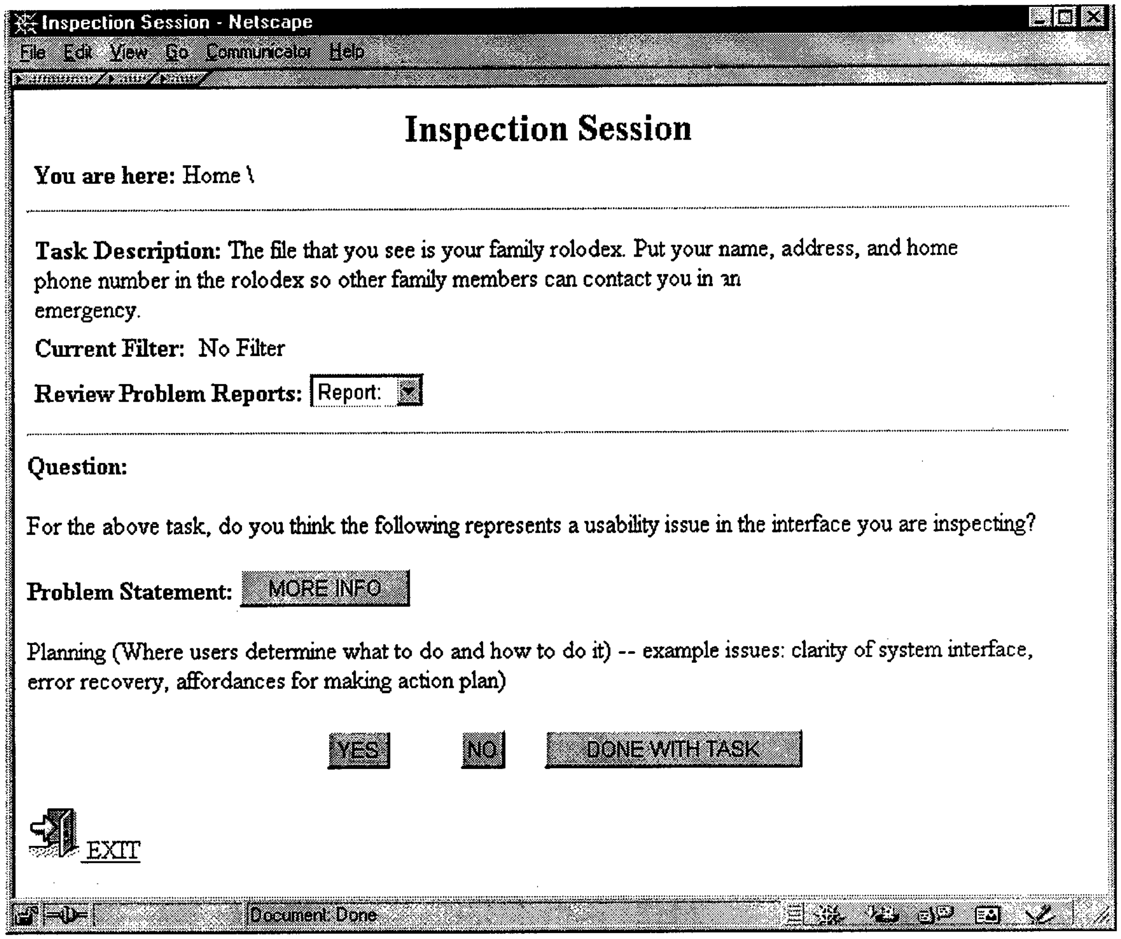
Please select the appropriate task and filter:

Task Name: Task 1: Insert a record

Description: The file that you see is your family rolodex. Put your name, address, and home phone number in the rolodex so other family members can contact you in an emergency.

Filter: No Filter

FIGURE 3-9. Inspection Session Setup Screen.


Inspection Session - Netscape
File Edit View Go Communicator Help

Inspection Session

You are here: Home \

Task Description: The file that you see is your family rolodex. Put your name, address, and home phone number in the rolodex so other family members can contact you in an emergency.

Current Filter: No Filter

Review Problem Reports: Report:

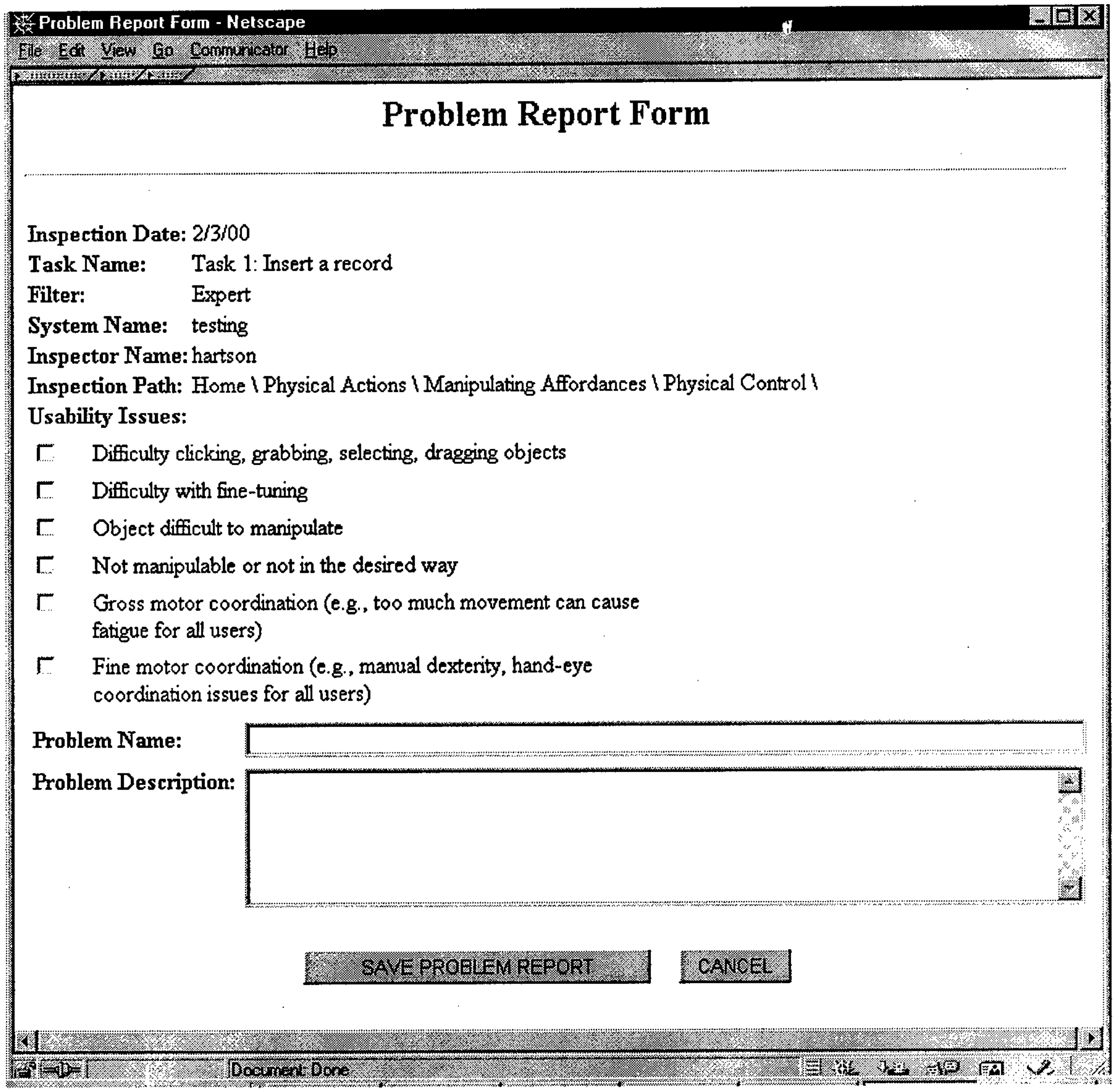
Question:

For the above task, do you think the following represents a usability issue in the interface you are inspecting?

Problem Statement: MORE INFO

Planning (Where users determine what to do and how to do it) -- example issues: clarity of system interface, error recovery, affordances for making action plan)

FIGURE 3-10. Inspection Session Main Screen with Problem Statement Description.


Problem Report Form - Netscape
File Edit View Go Communicator Help
SAVE PROBLEM REPORT
CANCEL
Document: Done

Problem Report Form

Inspection Date: 2/3/00
Task Name: Task 1: Insert a record
Filter: Expert
System Name: testing
Inspector Name: hartson
Inspection Path: Home \ Physical Actions \ Manipulating Affordances \ Physical Control \
Usability Issues:

- □ Difficulty clicking, grabbing, selecting, dragging objects
- □ Difficulty with fine-tuning
- □ Object difficult to manipulate
- □ Not manipulable or not in the desired way
- □ Gross motor coordination (e.g., too much movement can cause fatigue for all users)
- □ Fine motor coordination (e.g., manual dexterity, hand-eye coordination issues for all users)

Problem Name:

Problem Description:

FIGURE 3-11. Problem Report Form.

Focused Inspection Process

Focusing of the inspection is significantly different from the cognitive walkthrough where the evaluator has to provide their own interpretation as to how "deep" they should investigate a particular area. As a result, the UPI is expected to be a much more efficient approach to inspection than the cognitive walkthrough by providing a pre-defined knowledge base of usability concepts and issues, arranged in a hierarchical framework.

Built-in Classification of Problems

Effective problem reporting demands accurate, complete and unambiguous communication of usability problem descriptions to the cognizant individuals within the development organization. Unfortunately, most expert-based insp ection methods do little more than provide a list of usability problems that are often vague, imprecise, or incomplete. As usability evaluators have no framework or systematic method to guide and structure the capture and reporting of usability problem data, much of the originally available information is lost. The UPI, however, provides the means to document problems via a shared classification structure with the Usability Problem Classifier. This shared structure helps the inspector to take the discussion beyond the interaction feature itself (e.g., an ineffective button label) to a consideration of the effect on the user in a task context and the cause and cure in the interaction design. Thus, it is not enough to say something is wrong with a given feature (e.g., a bad button label). The UPI helps the inspector to build a more complete explanation: the label fails to give adequate cognitive affordance because it does not clearly indicate the functionality behind the button so that users may not be able predict the outcome of clicking on that button. Classification of problems offers a significant advantage over the heuristic evaluation method where inspectors often have difficulty assigning a specific heuristic to a very complex interaction problem having both feedback and error recovery aspects.

Considers Error Performance and Physical Usability Issues

Although certain aspects of the UPI (e.g., the use of goals, tasks, and intentions) may appear to be similar to cognitive models such as GOMS (Card et al., 1983), the UPI is quite different in terms of user behavior. Cognitive models, such as GOMS, assumes expert user and error-free behavior; an assumption not required for using the UPI. GOMS essentially examines issues with Physical Actions, missing the critical usability issues associated with Planning and Assessment. Because of the assumption of error-free behavior, GOMS does not lead directly to usability problem identification. Also, unlike the cognitive walkthrough, the UPI allows the evaluator to consider significant physical usability issues (e.g., Fitts' law, object design, disability accommodation) in addition to the traditional cognitive problems.

CHAPTER 4. PILOT STUDY OF THE UPI TOOL

To test the initial concepts of the UPI, and the Interaction Cycle and UAF it is based on, a network communication company provided an opportunity to evaluate a commercial message management service. The message management service allows voice access to email using voice recognition and synthesis technology. The message management service is intended to help traveling professionals gain access to email when it is impractical or impossible to connect a laptop to a data port. In addition, the message management service also provides a Web site where the user can setup features for pre-specified replies, maintain email addresses, prioritize incoming messages, and poll existing email accounts.

The goal of the usability evaluation was to provide a rapid and early, expert evaluation of the voice and Web interface for the message management service. The developer was interested in receiving problem descriptions as quickly as possible after the evaluation in order to rapidly change the system interface before a full product release was initiated. A secondary goal of the evaluation was to provide an opportunity to test the UPI by comparing it with the traditional heuristic method (i.e., discount usability engineering) commonly used for expert evaluation.

An important issue considered in the study was the number of evaluators given two separate goals: one to meet the rapid evaluation goal of the developer and the other goal for testing the UPI. Six evaluators were necessary to meet both goals, with three evaluators assigned to the UPI method and three evaluators to the heuristic method. Reasons for limiting the number of evaluators were very similar to those documented in Dutt, Johnson, and Johnson (1994). First, time to complete the evaluation was a factor since the system was near a product launch date. Second, the number of evaluators satisfies the bounds recommended by Nielsen and Molich (1990) for performing heuristic evaluation (i.e., 3-5 evaluators). Third, Nielsen and Landauer (1993) argued that one would rarely evaluate a single user interface to the 'bitter end' without applying iterative design to fix the usability problems found by the first few evaluators. Fourth, Nielsen and Landauer point out that the highest ratio of benefits to costs for medium-large software projects is for 3.2 test users and 4.4 heuristic evaluators. Finally, Virzi (1992) showed the likelihood of uncovering a new usability problem decreases as more and more subjects participate in a usability evaluation.

Experience level of the evaluators was another important consideration of this evaluation. The developer of the system did not have personnel trained in usability evaluation. Evaluators came from the university where they had both formal instruction and some experience with cognitive aspects of users, software design, and usability evaluation. In addition, expert evaluators would be fairly naive with respect to the domain of the system since the message management service was a new product. As a result, assistance in using the system was necessary as documented by Nielsen (1994b). With input from the developer, developing representative scenarios for both the web and voice interfaces were easily accomplished within a short amount of time. These scenarios included both a goal and task description.

In addition to number of evaluators and experience level, time to complete the evaluation was another concern. To complete the evaluation and quickly turn around the results, the task scenarios were restricted so that evaluators could easily complete the training and evaluation in one three-hour session. Since the developer was interested in specific areas of the system, a task-based approach was used for both methods rather than including an additional "free-form" exploration. The limited time for the evaluation session is in-line with recommendations provided by Dumas, Sorce, and Virzi (1995), where it appears to be more effective to have a greater number of evaluators work for a shorter time than to have one or two evaluators examine an interface for an extended period of time.

METHOD

Participants

A total of six graduate students from the Industrial and Systems Engineering Department participated as evaluators. These graduate students had comparable levels of formal training in human-computer interaction and usability evaluation methods. All evaluators also had formal academic exposure to usability heuristics, but none had experience with the UPI. Three evaluators were assigned to the UPI technique and the other three were assigned to the heuristic technique, while roughly balancing the experience level of each group.

Materials and Equipment

The evaluation used a fully featured prototype of the message management service. The message management service included a Web interface and a voice interface. In consultation with the developer, three representative usage scenarios were developed for both interfaces. Only one scenario with three distinct tasks was developed for the Web interface since the primary goal from the developer was to focus on the voice interface. Two scenarios, one with four tasks and the second with two tasks, were developed for the voice interface. The respective evaluation groups used Nielsen's revised set of ten heuristics (Nielsen, 1994a) and a Web-based version of the UPI. The early version of the UPI used an Interaction Cycle with four major parts: (1) Planning, (2) Physical Actions, (3) Assessment, and (4) Independent. The first three parts of the Interaction Cycle followed the description provided in Chapter 3. The Independent part of the Interaction Cycle included issues not specifically related to task-based usability problems. Overall interaction complexity and consistency are some examples of issues related to the Independent area. The Outcome part of the System Interaction Cycle was not used in this pilot study as it was still under formative development. Evaluators used on-line problem report forms to document identified problems. Using these forms, evaluators could provide a description of each problem, and classify it according to a specific heuristic or UPI problem type.

Procedure

During the first hour of the evaluation, each expert received equal amounts of training in their assigned method. The training was a review for the heuristic evaluators since they had been exposed to heuristics in their course work. Evaluation of the message management service began by providing each expert evaluator with a scenario description, a goal statement, and a specific task description. Both evaluator groups were free to accomplish the tasks in a way that seemed reasonable to them and to note possible problems with the interface while accomplishing the tasks. The UPI evaluators learned to consider the user's possible intentions and action sequences when using the Web-based inspection tool. Intention and action sequence information was not formally included as part of the training for heuristic evaluators since it is not a common approach when doing heuristic evaluation. All evaluators had two hours to complete the evaluation. At the end of the study, each evaluator also rated problem severity on a four-point scale adapted from Nielsen (1994a). Evaluators used a composite list of all usability problems to provide severity ratings.

RESULTS

Problem Identification

Each expert evaluator produced a list of problems with respect to the evaluation technique used. All problems listed condition (i.e., heuristic vs. UPI), scenario, and problem type identifiers, and combined into one list. Evaluators identified a total of 40 unique problem types after removing duplicates. Figure 4-1 shows the general distribution of different problem types identified by each group. As shown, both methods identified a common set of 10 problems. A chi-square test comparing the heuristic method to the UPI method revealed no significant difference (p >.10) in the total number of problem types in each condition (heuristic 26 vs. UPI 24) or in the number of unique problem types (heuristic 16 vs. UPI 14).

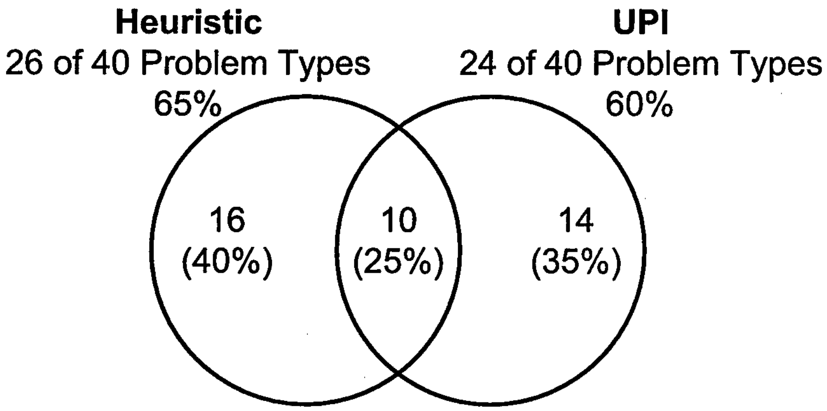


FIGURE 4-1. Comparison of Number of Unique Problem Types Identified in Heuristic and UPI Methods.

Based on the 40 unique problem types, the mean number of usability problem instances found by a single evaluator was 11.2 (28%). One evaluator found as many as 14 usability problems (35%); whereas, another evaluator found only 8 usability problems (20%). The mean number of usability problems found by a heuristic evaluator was 12 (30%), compared to 10.3 (26%) for a UPI evaluator.

Content Analysis

Evaluators classified each problem found using the respective evaluation technique. For the heuristic method, the evaluator assigned each problem to one of Nielsen's (1994a) heuristics. The highest percentages of heuristics classified by the evaluators was Visibility of System Status (33%), Consistency and Standards (17%), and Match Between System and the Real World (11%). In addition, some evaluators classified problems into an "other" category (11%) when they did not fit into one of the ten heuristics. These problems involved voice recognition and the system not responding as expected. Heuristic evaluators selected the heuristic most closely related to the type of problem identified, but all evaluators also noted additional heuristics that could apply. Of the 26 problems identified, heuristic evaluators identified 8 as possibly violating at least one other heuristic in addition to the primary heuristic. The percentage of problems identified by evaluators using the UPI were Planning (58%), Physical Actions (16%), Assessment (16%), and Independent (10%). None of the problem classifications by UPI evaluators overlapped across Interaction Cycle parts, and additional categories were not needed for complete classification.

Classifying Problems

Six of the problem types in the heuristic method were shared with at least one other evaluator. Looking closer at these problem types, evaluators classified three of the problems as the same heuristic. The other three problems were classified differently by at least one of the evaluators finding that same problem. Five of the problem types in the UPI method were shared with at least one other evaluator. Four of the five problems were classified at the exact same point in the UPI (i.e., Planning and Physical Action parts). All three evaluators classified one problem within the Assessment part, but disagreed on the final classification within Assessment.

Comparison of Unique Problems

Examination of the non-overlapping unique problems identified by each method provides some insight into the difference between the heuristic and UPI methods. The 14 unique problems identified by the UPI method primarily involve Planning and Physical Action issues. As an exercise, an HCI expert attempted to assign heuristics to each of these 14 unique problems and found it very difficult to assign heuristics to problems that were about determining an intention or translating an intention to an action description. The difficulty was noted in the need to use more than one heuristic to describe a particular problem found in the UPI. Problems with missing features and recognizing different modes were especially difficult to assign to a heuristic.

Similarly, the HCI expert looked at the 16 unique problems identified by the heuristic evaluation. Seven of these sixteen problems were about some kind of feedback issue (e.g., Visibility of System Status) and were easy to identify as Assessment problems in the UPI. The remaining 9 problems involved heuristics such as Consistency and Standards, Recognition Rather than Recall, Error Prevention, and Aesthetic and Minimalist Design. These heuristics map to many different parts in the UPI because they can be issues at every part in the UPI and need the Interaction Cycle context at time of classification. Consistency and Standards, for example, are lumped into one heuristic, but many different kinds of consistency and standards apply to different aspects of interaction and different task-related situations. The UPI captures these differences throughout the various parts of the Interaction Cycle.

Severity Analysis

All six evaluators rated severity of the 40 unique problem types on a scale from 1 (cosmetic problem) to 4 (usability catastrophe). Kendall's coefficient of concordance between the six evaluators was $\underline{W}$=.41, which is statistically significant, $\chi^2(39) = 96.9$, $\underline{p} < .01$. This finding indicates the 40 problem types were, indeed, different from each other and that evaluators had better than random agreement in their classifications. The median rating of severity for all 40 problems was 2.0. Median ratings of severity were highest for the 10 problems identified by both methods (median = 3.0) and the 14 unique problems identified by the UPI evaluators (median = 3.0). The 16 unique problems identified by heuristic evaluators resulted in a median rating of 2.0. Severity was also examined by splitting problems into most severe (median of 3.0 or higher) and least severe (median of 2.50 or lower). The heuristic method isolated primarily the least severe problems; whereas, the UPI method found significantly more severe problems, as shown in Table 4-1, $\chi^2(2) = 7.36$, $\underline{p} < .05$.

TABLE 4-1. Number of Problems Found by Severity and Method.

	UPI Unique	**Heuristic Unique**	**Shared**
Most severe	7	2	6
Least severe	7	14	4

DISCUSSION

As an initial validation of the UAF concept and its UPI tool, this study focused on discovering the differences between the UPI and heuristic methods of usability evaluation. In particular, the desire was to find out if the UPI could be used as easily as the heuristic method. The initial analysis suggests the two methods are similar in total number of problem types identified, yet different in the types of unique problems found, with the heuristic method finding a greater number of the minor problems. This finding is consistent with other research involving severity analysis with the heuristic method (Doubleday et al., 1997; Jeffries et al., 1991).

The relatively low mean number of problems found by a single evaluator in either category confirms the finding that one cannot rely on a single person to perform an expert evaluation (Nielsen & Molich, 1990). The low overall rate of detection (28%) is possibly attributable to the voice interface -- a focus of two-thirds of the evaluation. Nielsen and Molich found detection rates of 26% and 20% for two different voice systems and noted that voice systems have an extremely low persistence, giving evaluators less opportunity to ponder the details.

Particularly noteworthy is the distribution of problems around the Interaction Cycle, with the majority occurring in the Planning part. This finding is similar to the results found by Cuomo and Bowen (1992, 1994), who also applied Norman's theory of action and found the majority of problems in the action specification stage. As expected, a great number of problems were found in the Planning part since users were not experts with the system and needed help from the interface to make the first translation from an intention to an action description. For new users, the first translation for a task is generally the most difficult and it behooves the interaction designer to provide appropriate affordances to support the user in making the translation. If designers can help users at this point, the rest of the Interaction Cycle has a greater chance of succeeding.

Evaluators using the UPI learned how to separate out characteristics of usability problems based on the Interaction Cycle parts. Planning, Physical Actions, and Assessment parts of the Interaction Cycle helped evaluators think of problems related to the tasks they performed with the interface. Evaluators classified only 10% of the problems in the Independent part of the Interaction Cycle, indicating this area might be less relevant when trying to differentiate interface issues that generally have a task component.

The results concerning unique problems identified by each method are similar to those of other studies where heuristic evaluation is compared to a method such as the cognitive walkthrough (e.g., Cuomo & Bowen, 1992; Cuomo & Bowen, 1994; Desurvire, 1994; Desurvire et al., 1992; Dutt et al., 1994; Jeffries et al., 1991; Sears, 1997). The UPI seems more likely to find problems similar to those found by the cognitive walkthrough since it, too, is built upon an interaction-based approach. However, unlike the cognitive walkthrough, evaluators can use the UPI as a rapid evaluation method much like heuristic evaluation and can discover usability problems associated with physical actions. UPI evaluators were able to learn the method easily with only one hour of training. In addition, UPI evaluators were able to complete the evaluation scenarios within the allotted two-hour time.

Finally, the UPI offers a potential advantage over other evaluation methods since classification is built into the tool. Because the UPI shares the common structure of the UAF, evaluators are able to both consider potential usability issues and provide complete classification during the same evaluation session. For other methods like the cognitive walkthrough and heuristic evaluation, classification usually occurs after the evaluation and often by another person (e.g., Cuomo & Bowen, 1992; Cuomo & Bowen, 1994; Nielsen & Molich, 1990). Such an approach has two associated costs. First, information can be lost between the time a specific evaluator describes a problem and an additional evaluator classifies that problem. Second, many development efforts may not have the resources to support a separate classification activity.

CHAPTER 5. RELIABILITY STUDY

Usability support tools offer the most value to interaction development groups if they are used consistently and predictably, from practitioner to practitioner. Without this kind of usage reliability, one evaluator using a usability tool can get one result and another a different result, and the usability data for the project will depend on the individual using the tools. Although the reliability of a method appears to be inherently important, the literature on UEM reliability studies is essentially non-existent. Thus, more research is needed to establish the reliability of various UEMs.

The purpose of this reliability study was to determine if the UAF showed significantly better than chance agreement when usability experts classified a given set of usability problem case descriptions using the framework as a classification tool. In order to make substantive conclusions about the level of reliability, results from the UAF reliability study are also compared to the reliability documented in the Usability Problem Taxonomy (Keenan et al., 1999) and a heuristic reliability study obtained from the same experts using Nielsen's (1994a) revised set of heuristics.

METHOD

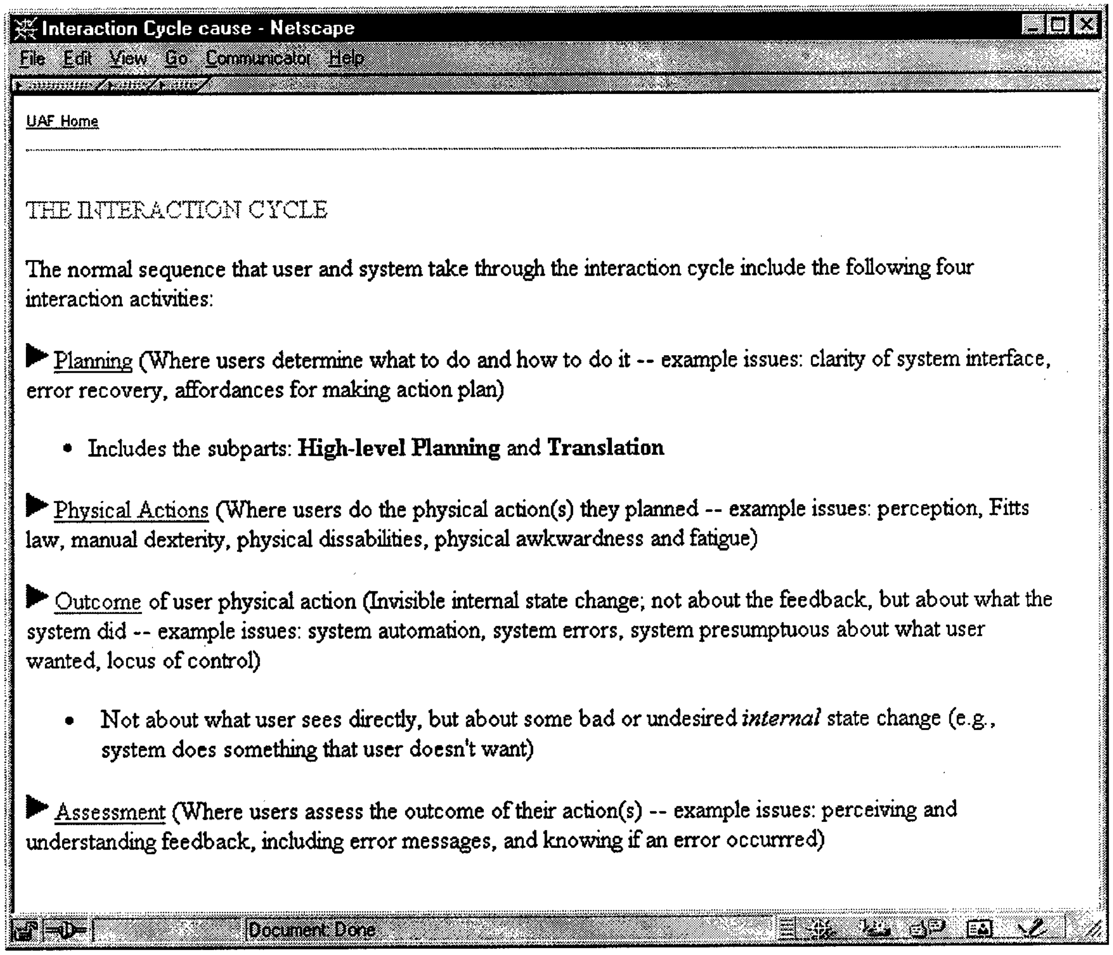
Participants

The participants for the UAF reliability study consisted of ten usability professionals recruited from government and commercial organizations. Nine of the ten usability professionals participated in the follow-up heuristic reliability study. The participants came from organizations where usability engineering (design, test, or evaluation) was a formal part of their daily experience. All participants possessed at least a bachelor's degree in computer science, human factors, psychology, or industrial engineering. A majority (6 of 10) of the participants possessed an advanced degree (masters or PhD). The average age of the participants was 35 years, ranging from 25 to 47 years. All participants had a minimum of three years experience in user interface design, test, and/or evaluation (M = 7.9). Participants were equally split in terms of their self-reported usability specialty with half coming from the design perspective and the other half coming from the test and evaluation perspective.

Materials

Materials for the reliability study included a local Website containing the UAF content linked together to facilitate traversal of the knowledge base. Figure 5-1 shows the UAF start page with the four primary areas of the Interaction Cycle (Planning, Physical Actions, Outcome, and Assessment) represented as hypertext links. This version of the UAF did not include an Independent part since results from the pilot study did not show it to add much value in the description process (reference Chapter 4).

Figure 5-2 shows the displayed result of selecting on the Physical Actions link where the participant is provided the detail for this next level in the UAF. Each link provided the participant with downstream visibility into the next level of the UAF. Thus, Figure 5-2 shows the first level of detail for the Physical Action part of the Interaction Cycle.


Interaction Cycle cause - Netscape
File Edit View Go Communicator Help
Document: Done

UAF Home

THE INTERACTION CYCLE

The normal sequence that user and system take through the interaction cycle include the following four interaction activities:

▶ Planning (Where users determine what to do and how to do it -- example issues: clarity of system interface, error recovery, affordances for making action plan)

- Includes the subparts: **High-level Planning** and **Translation**

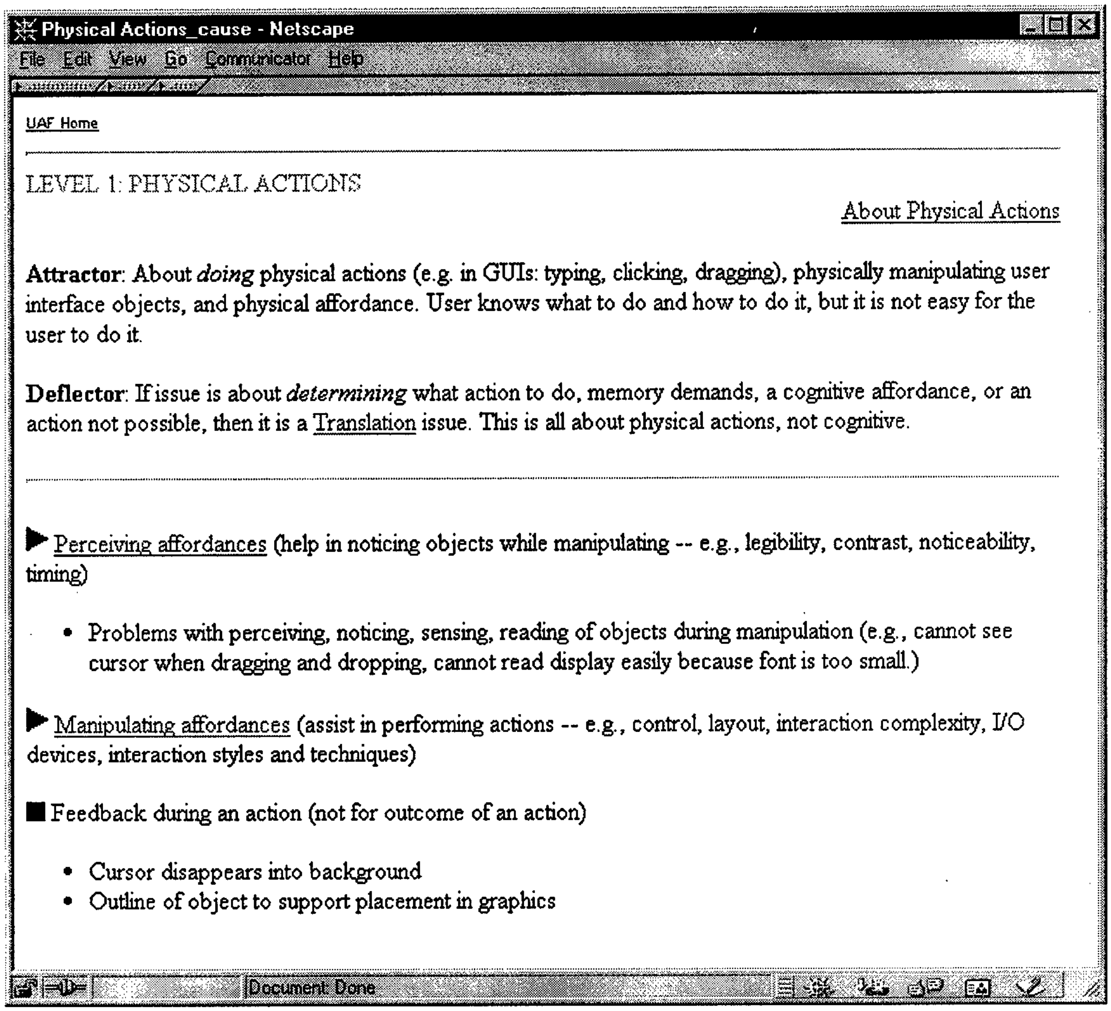
▶ Physical Actions (Where users do the physical action(s) they planned -- example issues: perception, Fitts law, manual dexterity, physical dissabilities, physical awkwardness and fatigue)

▶ Outcome of user physical action (Invisible internal state change; not about the feedback, but about what the system did -- example issues: system automation, system errors, system presumptuous about what user wanted, locus of control)

- Not about what user sees directly, but about some bad or undesired *internal* state change (e.g., system does something that user doesn't want)

▶ Assessment (Where users assess the outcome of their action(s) -- example issues: perceiving and understanding feedback, including error messages, and knowing if an error occurred)

FIGURE 5-1. Start Page for The UAF.


Physical Actions_cause - Netscape
File Edit View Go Communicator Help
Document: Done

UAF Home

LEVEL 1: PHYSICAL ACTIONS

About Physical Actions

Attractor: About *doing* physical actions (e.g. in GUIs: typing, clicking, dragging), physically manipulating user interface objects, and physical affordance. User knows what to do and how to do it, but it is not easy for the user to do it.

Deflector: If issue is about *determining* what action to do, memory demands, a cognitive affordance, or an action not possible, then it is a Translation issue. This is all about physical actions, not cognitive.

▶ Perceiving affordances (help in noticing objects while manipulating -- e.g., legibility, contrast, noticeability, timing)

- Problems with perceiving, noticing, sensing, reading of objects during manipulation (e.g., cannot see cursor when dragging and dropping, cannot read display easily because font is too small.)

▶ Manipulating affordances (assist in performing actions -- e.g., control, layout, interaction complexity, I/O devices, interaction styles and techniques)

■ Feedback during an action (not for outcome of an action)

- Cursor disappears into background
- Outline of object to support placement in graphics

FIGURE 5-2. Example of Physical Actions Page in the UAF.

Fifteen usability problem case descriptions were selected from a larger database containing over 100 usability problem cases. These descriptions were collected from various software development projects, academic examples, and personal experience. Case descriptions employed in this study were limited to those representing only a single usability problem or concept; case descriptions representing multiple, related usability problems were not considered. The 15 case descriptions were also selected on the basis of their real-world expected frequency of occurrence relative to their location within the Interaction Cycle. Based on the pilot study discussed in Chapter 4, a majority of the usability problems found existed within the Planning portion of the Interaction Cycle. Assessment contained the second largest portion of the usability problems; the fewest problems found occurred within the Physical Actions portion of the Interaction Cycle. The pilot study did not investigate usability problems associated with the Outcome part of the Interaction Cycle since this part was still under formative development. Although this version of the UAF contained content for the Outcome part of the Interaction Cycle, none of the problem descriptions in this study related to Outcome-type issues. Table 5-1 provides a summary of the 15 cases used in the reliability study, and how they are distributed within the UAF. Materials for the heuristic reliability study consisted of Nielsen's (1994a) revised set of ten heuristics shown in Table 5-2 along with the same 15 usability problem descriptions used in the UAF study (Table 5-1).

TABLE 5-1. Usability Problems Used in the Reliability Study.

Case #	Type of Usability Problem	Relevant Area in UAF
1	Unreadable error message	Assessment
2	User does not understand master document feature	Planning (High-level)
3	User cannot find a feature to support re-using document numbers in a document retrieval system	Planning (Translation)
4	User clicks on wrong button	Physical Actions
5	User cannot directly change a file name in an FTP program	Planning (Translation)
6	User cannot tell if system is performing requested operation	Assessment
7	User wants to fix database error but is confused by button labels for appropriate action	Planning (Translation)
8	Program does not provide a Ctrl-P shortcut for printing	Planning (Translation)
9	User cannot understand error message provided by system	Assessment
10	Unusually long error message	Assessment
11	Unwanted confirmation message	Assessment
12	User does not see way to select odd font size	Planning (Translation)
13	Data entry format not provided	Planning (Translation)
14	Uncontrollable scrolling	Physical Actions
15	Vision-impaired user needs preference options for setting larger font size	Planning (Translation)

TABLE 5-2. Revised Set of Usability Heuristics (from Nielsen, 1994a).

Heuristic
Visibility of system status
Match between system and real world
User control and freedom
Consistency and standards
Error prevention
Recognition rather than recall
Flexibility and efficiency of use
Aesthetic and minimalist design
Help users recognize, diagnose, and recover from errors
Help and documentation

Procedure

Prior to the collection of data, participants signed the Informed Consent Form included in Appendix B. For the UAF reliability study, each participant viewed a 20-minute tutorial on the UAF. This tutorial involved an on-line description of the Interaction Cycle components, the structure of the UAF, and an example of how to use the Web-based UAF to classify a usability problem (Appendix C). Participants then read the 15 case descriptions and used the Web-based UAF to classify the problem. Even though problem descriptions were selected on the basis of having only one usability issue, participants were directed to classify only the primary problem on the chance they interpreted a problem description as having two or more usability problems. Participants could traverse all paths of the UAF before noting their final classification of the usability problem.

One month later, the same experts, less one, participated in the heuristic reliability study using the 15 usability problem case descriptions from the UAF reliability study. Participants received a heuristic evaluation packet in the mail with Nielsen's (1994a) revised set of ten heuristics, a 20-minute training package adapted from Nielsen's (1993) *Usability Engineering* book, and paper forms to record the primary heuristic they would apply to each usability problem description (Appendix C).

Hypotheses

Although a primary goal of the study was to document the reliability of the UAF, two hypotheses were based on formative evaluation. First, the UAF would result in a higher overall reliability score (kappa) than was found in previous work with the Usability Problem Taxonomy (Keenan et al., 1999). Keenan et al.'s work showed classification was reliable at the first classification level (the level of the five primary categories) on the artifact dimension ($\underline{\kappa} = .403$, $\underline{p} < .001$), but marginally reliable on the task dimension ($\underline{\kappa} = .095$, $\underline{p} > 0.10$). The expectation for higher agreement is based on extensive formative evaluation and on the assumption that the Interaction Cycle, based on Norman's (1986) theory of action model, provides a more natural way to think about the types of problems users encounter. In addition, the provision for alternative paths for some classification choices rather than a pure hierarchical structure allows for reconvergence on the same final classification node even though the users of the tool may take slightly different paths to the final end node.

The second hypothesis is the UAF will result in higher agreement scores than results obtained from the same experts using heuristics to classify the 15 usability problem cases. Although not originally intended as a classification framework, heuristics have proven to be important labels for both finding and discussing problems. As a result, knowing how reliable they are in classifying usability problems provides a valuable data point for both researchers and practitioners.

Data Collection and Analysis

Reliability Measure

Cohen's (1960) kappa statistic was used to examine observer agreement since the UAF is essentially a hierarchical structure of categorical information. Kappa (κ) is scaled between -1 and +1. Positive values of kappa correspond to greater than chance agreement, zero represents only chance agreement, and negative values correspond to less than chance agreement. Kappa is approximately normally distributed and can be used to test the null hypothesis of whether agreement exists beyond the chance level. Kappa traditionally assesses agreement between two observers. In the present study, more than two observers participated, thus requiring a modification to Kappa. Fleiss (1971) provides an extension to the kappa statistic to measure the level of agreement among several observers.

Scoring Expert Agreement

The primary data for UAF reliability study were the participant's path through the UAF in classifying a problem. For each case description, the researcher recorded the path taken by the participant and documented their selection of end-page descriptions. Figure 5-3 shows an example of this documentation process where a usability expert traverses the UAF to describe a usability problem involving a user having a difficult time reading a feedback message. Levels in the UAF represent the hierarchical structure of usability concepts and issues (reference Appendix A). At each level, or displayed page, the participant selects the most appropriate usability concept related to the problem description.

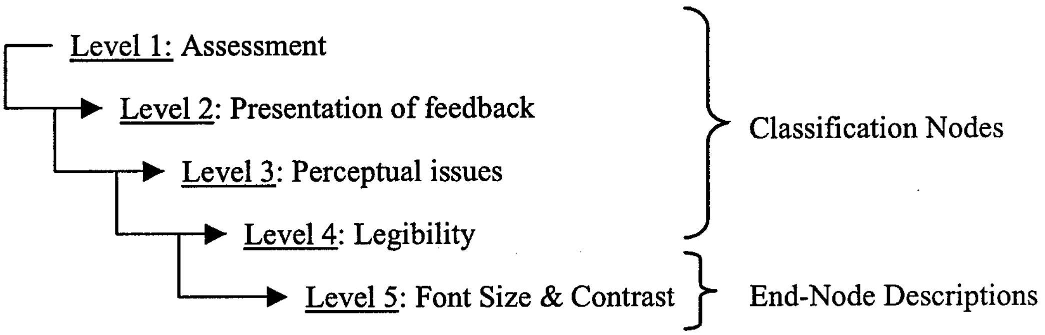

Level 1: Assessment
Level 2: Presentation of feedback
Level 3: Perceptual issues
Level 4: Legibility
Level 5: Font Size & Contrast
Classification Nodes
End-Node Descriptions

FIGURE 5-3. Example Path for a Usability Problem Involving a Feedback Message.

In the example shown in Figure 5-3, the first four levels comprise the classification nodes where choices were designed to be orthogonal. The lowest level of the hierarchy, Level 5 in this example, was not intended to be orthogonal for classification. Rather, these end-node descriptors were developed to augment the classification with a description of several possible causes related to the final classification node. Font size and contrast were the descriptors chosen to portray the nature of the legibility problems in the above example. Agreement at the end-node descriptions was defined as two or more experts having an element in common. For example, if one expert selected *Size and Color* while another expert selected *Size and Contrast*, then these two experts are in agreement since *Size* is a common element.

Because the UAF is comprised of a number of classification nodes at various levels (as many as 6 levels), agreement was calculated at each of the different levels within the hierarchical structure as well as overall agreement at all end-node descriptions. For each usability case description, the participant using the UAF is presented with a range of choices that are dependent upon the path taken to describe the problem. At the top levels of the UAF, the number of choices are usually small; typically the choices are between two or three items. The UAF broadens at deeper levels, presenting the user with as many as eight choices at the lowest classification nodes. Therefore, the small differences in choices made early on result in large differences in terms of the number and kinds of choices faced later. As a result, the hierarchal structure of the UAF essentially holds up a higher standard for reliability because once two classifiers disagree, there is little or no chance for them to later reconverge to agreement.

Data from the heuristic reliability study were relatively straightforward in terms of measuring agreement since there were only ten categories and no hierarchical levels. The nine participants in the heuristic reliability study indicated their primary choice of a heuristic that applied to each case description on paper forms where Nielsen's (1994a) ten heuristics were listed.

RESULTS

Reliability measures, such as kappa, are intended to measure classifier agreement across a fixed number of categories. Classifier agreement in the UAF was analyzed in three ways: (1) reliability at each level within the hierarchical structure, (2) reliability within the respective parts of the Interaction Cycle (i.e., Planning, Physical Actions, Assessment), and (3) overall reliability for end-node descriptions. Classifier agreement for the heuristic reliability study was calculated for the ten heuristics as one level of end-node descriptions.

Agreement at Levels in the UAF

Table 5-3 shows an example of the data from one usability case description. Case 10 was about an unnecessarily long error message displayed to the user. Level 1 shows that all 10 participants agreed that this particular usability case description was an *Assessment* problem because it involved a feedback message. At the next level (Level 2), 9 of the 10 participants agreed the case description was about the *Content* of the feedback message. One participant felt the issue was related to the *Existence* of the feedback message. To continue measuring agreement accurately, the data from participant #6 was eliminated from further reliability measures since this participant was now taking a different path than the remaining nine. Thus, at Level 3, 8 of the remaining 9 participants agreed that the issue was about the *Clarity* of the feedback message. Participant #9 felt the problem was related to the *Completeness* of the feedback message. As a result, the data from participant #9 was eliminated from reliability analysis at the next level. At Level 4, all 8 remaining participants agreed that the issue was about the *Complexity* of the feedback message. At the end page for this path (Level 5), all 8 participants selected *Volume & Verbosity* as the usability cause for this case description. The example illustrated in Table 5-3 shows the approach for calculating reliability at different levels by eliminating participants that proceeded down a different path from the majority. This helped to avoid continuous penalties for disagreement at lower levels when a participant was on a different path and had no opportunity to see the same choices as the other participants.

TABLE 5-3. Example Summary of Participant Categorization of a Usability Case Description.

CASE 10	**LEVEL IN UAF**				
Participant	LEVEL 1	LEVEL 2	LEVEL 3	LEVEL 4	LEVEL 5
1	Assessment	Content	Clarity	Complexity	Volume & Verbosity
2	Assessment	Content	Clarity	Complexity	Volume & Verbosity
3	Assessment	Content	Clarity	Complexity	Volume & Verbosity
4	Assessment	Content	Clarity	Complexity	Volume & Verbosity
5	Assessment	Content	Clarity	Complexity	Volume & Verbosity
6	Assessment	Existence	~~Level inappropriate~~	----------------	--------------------
7	Assessment	Content	Clarity	Complexity	Volume & Verbosity
8	Assessment	Content	Clarity	Complexity	Volume & Verbosity
9	Assessment	Content	Completeness	~~Level of detail~~	--------------------
10	Assessment	Content	Clarity	Complexity	Volume & Verbosity
Agreement	**10 out of 10**	**9 out of 10**	**8 out of 9**	**8 out of 8**	**8 out of 8**

Results of reliability calculations across all case descriptions appear in Table 5-4. Column 2 indicates the number of cases analyzed for each level within the UAF. Depending on the case, participants had to traverse a number of hierarchical levels before reaching the final page with end-node descriptions. For example, some cases used in this study required navigation down to only the fourth level in the UAF before end-node descriptions were presented. As shown in Table 5-4, 9 cases required Level 5 classification while only 3 cases required Level 6 classification. Values in the $\mathbf{P_0}$ column indicate the proportion of observed agreement while values in the **P_c** column indicate the proportion of agreement expected by chance. Kappa accounts for the fact that the proportion of chance agreement decreases as the number of choices increase. As shown in Table 5-4, the proportion of chance agreement is higher at the top levels in the UAF than the lower levels because there are fewer choices at the top of the framework. Thus, observed agreement requires substantially higher values to overcome chance agreement at the top levels of the UAF. The kappa values shown in Table 5-4 (κ column) indicated strong agreement at all levels within the UAF, especially at the top levels of the framework. The **Z** column contains the observed values for the standard normal variate obtained by dividing kappa by its standard error. The high z values indicated that kappa scores were significantly greater than chance agreement ($p < .001$). For Level 6, kappa was not calculated since only 3 cases were relevant at this level, limiting the number of data points for consideration. With such few cases at Level 6, the agreement score would likely not be valid since the approximate normality assumption for kappa would be violated.

TABLE 5-4. Results of Reliability Analysis at Each Level in the UAF.

Level	**Cases at this level**	**P_o**	**P_c**	**κ**	**Z**
1	15	.987	.408	.978	20.25***
2	15	.979	.274	.972	22.17***
3	15	.800	.081	.783	60.77***
4	14	.781	.082	.762	53.28***
5	9	.752	.118	.719	32.37***
6	3	--	--	--	--

Note. Dashes indicate too few data points to calculate classifier agreement.
***$p < .001$

Agreement for the Interaction Cycle Parts of the UAF

Table 5-5 shows the reliability analysis for the Interaction Cycle parts of the UAF. The sub-part, Translation, was included because of its relevance to a number of usability problem case descriptions. The number of categories for the participant to select from at each part in the UAF is shown in column 2 of Table 5-5. The Interaction Cycle parts presented participants with 2 or 3 choices, except for Translation, which presented 5 choices. Column 3 shows how the case descriptions were distributed among the Interaction Cycle parts. The Planning part, which included High-Level Planning and Translation sub-parts, included the majority of cases (i.e., 8), followed by Assessment with 5 and Physical Actions with 2 relevant cases. Agreement was very strong for the Interaction Cycle parts with kappa ranging from .673 to .943. Agreement scores were significantly greater than chance as indicated by high z values ($p < .001$). The results also revealed that the Translation sub-part was more difficult in terms of obtaining consistent agreement among the participants. Planning and Assessment parts showed very high agreement, indicating that participants were able to easily differentiate problem attributes based on these two parts of the Interaction Cycle. Reliability calculations for classification within the Physical Actions part of the UAF were not possible because of the limited data points generated from only two relevant cases, again due to validity problems with the approximate normality assumption

TABLE 5-5. Results of Reliability Analysis for the Interaction Cycle Parts of the UAF.

Interaction Cycle Parts	**Categories**	**Relevant cases**	P_o	P_c	κ	**Z**
Planning	2	8	.987	.779	.943	3.65***
Translation	5	7	.760	.265	.673	17.92***
Physical Actions	2	2	--	--	--	--
Assessment	3	5	.960	.361	.937	15.33***

Note. Dashes indicate too few data points to calculate classifier agreement.
***$p < .001$

Overall Agreement

Overall agreement of classifiers was also calculated by examining the final end-node descriptions across all usability cases. The overall agreement provides reliability information for the various paths taken by each classifier. Kappa results for overall agreement (Table 5-6) showed strong reliability ($\kappa = 0.583$, $p < .001$), indicating agreement is greater than what would be expected by chance. In calculating kappa across all cases, the UAF is essentially transformed from six hierarchical levels into a flat structure with more than 150 end-node descriptions. Therefore, the probability of chance agreement was extremely small ($P_c = .048$) considering the number of possible end-node descriptions available to the classifiers.

TABLE 5-6. Overall Reliability for the UAF.

Number of Cases	$\mathbf{P_o}$	$\mathbf{P_c}$	$\boldsymbol{\kappa}$	**Z**
15	.603	.048	.583	61.86***

***p < .001

Heuristic Evaluation Results

Data from reliability calculations for the heuristic reliability study are shown in Table 5-7. Kappa results showed moderate agreement (κ = 0.325, p < .001), indicating agreement is greater than what would be expected by chance. Results from hypothesis testing for independent samples revealed that the reliability of the heuristic classifiers was not as strong as the reliability obtained from the UAF classifiers (p < .001), even reliability comparisons to the lowest classification nodes. Table 5-8 summarizes the results from the statistical hypothesis testing using the standard normal distribution (z) as the test statistic.

TABLE 5-7. Results of Reliability Analysis for Heuristic Reliability Study.

Number of Cases	$\mathbf{P_o}$	$\mathbf{P_c}$	$\boldsymbol{\kappa}$	**Z**
15	.404	.116	.325	19.13***

***p < .001

TABLE 5-8. Summary of Reliability Comparison between Heuristic and UAF Participants.

$\boldsymbol{\kappa}$ (UAF)	**$\boldsymbol{\kappa}$ (HE)**	**Z ($\boldsymbol{\kappa_{UAF} - \kappa_{HE}}$)**	**Conclusion**
.583 (Overall)	.325	12.90***	User Action Framework > Heuristic
.978 (Level 1)	.325	12.75***	User Action Framework > Heuristic
.972 (Level 2)	.325	13.79***	User Action Framework > Heuristic
.783 (Level 3)	.325	21.12***	User Action Framework > Heuristic
.762 (Level 4)	.325	19.54***	User Action Framework > Heuristic
.719 (Level 5)	.325	14.02***	User Action Framework > Heuristic

***p < .001

DISCUSSION

Built as a structured knowledge base of usability concepts and issues, the UAF is intended to provide a framework underlying usability engineering support tools to aid practitioners with a standardized method for developing usability problem descriptions that distinguish different problem types and help form a shared understanding of the specific attributes of the problem. The reliability study focused on determining the degree of consistent use by usability practitioners classifying a given set of usability problems. Consistent classification of usability problems is necessary to produce high quality problem reports that lead to more direct solutions and more efficient use of resources in the documentation process

Results from the UAF reliability study showed higher overall agreement ($\underline{\kappa} = .583$) than was found in previous work with the Usability Problem Taxonomy ($\underline{\kappa} = .403$). More importantly, agreement scores at all classification levels (Levels 1 – 5) within the UAF ($\underline{\kappa} = .719$ to .978) were higher than the top level of the Usability Problem Taxonomy. Finally, agreement using the UAF was significantly stronger than the results obtained from the same experts using the heuristic evaluation ($\underline{\kappa} = .325$). Evaluators using the UAF were especially consistent at using the parts of the Interaction Cycle to begin their classification of each usability problem. Only one evaluator on one usability problem diverged to a different part of the Interaction Cycle during the classification process. Such a result supports the notion that a model-based framework is important to providing a reliable classification system that helps build a shared understanding of the different attributes of a usability problem.

As a hierarchically structured knowledge base, the UAF provided much more description and discrimination power than the heuristic evaluation technique. Heuristic categories are generally not distinct and often result in evaluator confusion when selecting appropriate labels for a problem (Doubleday et al., 1997; Jeffries et al., 1991). In terms of measuring reliability, a hierarchically structured framework is more problematic than a flat structure such as Nielsen's (1994a) heuristics. In addition to overall reliability, classifier agreement at each level provided information as to how users were able to consistently understand and select choices as they traversed the hierarchical structure. In fact, reporting classifier agreement by level provided more information regarding the structure of the UAF than the information provided by the overall reliability score. Reporting reliability for only an end-node description hides valuable information about classifier agreement at previous levels reaching the end-node. As an example, consider the case shown in Figure 5-4 where evaluators start to disagree at Level 3. The circles represent the number of evaluators choosing a particular node in the classification hierarchy. Disagreement is apparent if reporting results at Level 4. However, to report on the agreement (or disagreement in this case) at Level 4 without consideration for the extent of agreement at higher levels essentially disconnects the framework from its designed purpose. That is, the description at Level 4 is only complete in the context of the path taken to a particular end-node description. In the example shown in Figure 5-4, the entire context of the path for 4 of the 10 classifiers is something like: *The usability problem is a high-level planning issue involving the user's model of the system in order to understand the overall concept.*

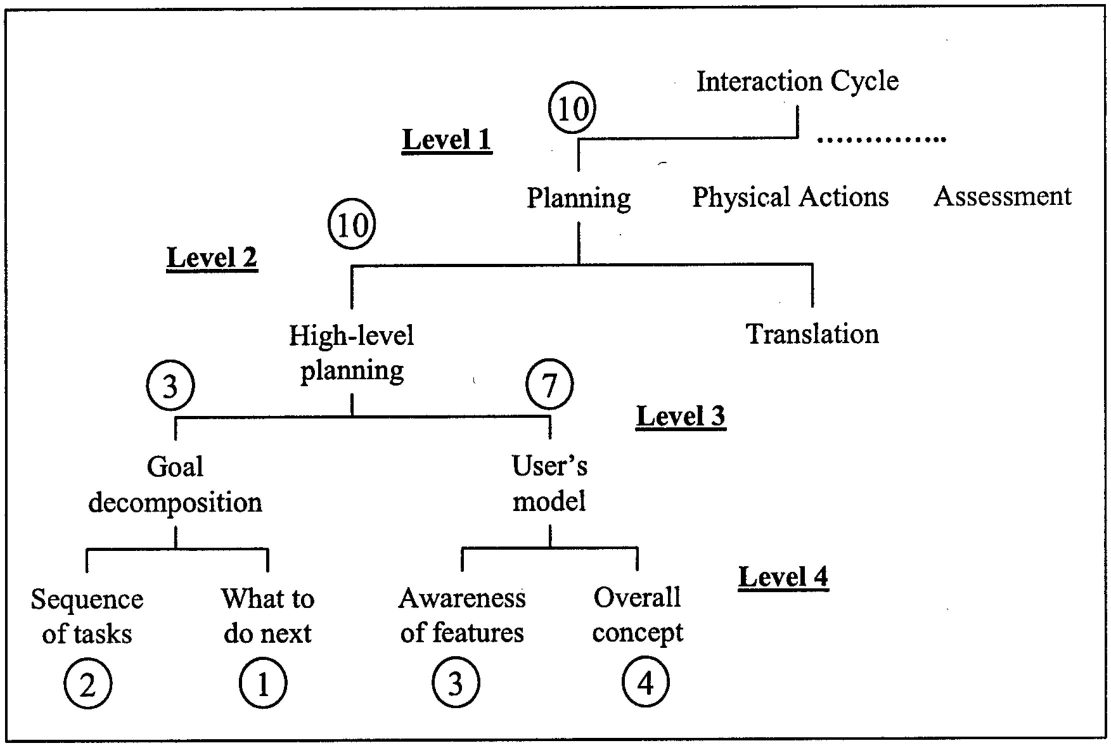


FIGURE 5-4. Example of Classification Path for a Usability Problem.

Although the results show improved reliability due to iterative refinements, literature-based comparisons are difficult to make. Reliability, as defined in this study, is not documented in the current literature on usability evaluation methods. Practitioners would not deny the importance of providing consistent results from usability engineering support tools, but the matter of operationally defining consistent performance is a different issue. Some practitioners may be interested in knowing that one evaluator can use a tool consistently across projects. Others may be more interested in knowing that different evaluators are relatively consistent in their use of the usability engineering support tool. In either case, the consistent use of a tool does not guarantee the output of an evaluation will produce quality problem reports that communicate problems and causes precisely, and suggest solutions for down-stream redesign activities. Providing better quality problem reports depends on both the structure of the usability framework that guides the description process and the content of the framework that helps to provide a complete understanding of the usability problem.

CHAPTER 6. UPI COMPARISON STUDY

The second step in evaluating the UPI was a comparative study of popular inspection methods using a lab-based usability test to baseline the set of real usability problems that impact users. In addition to the UPI, the heuristic evaluation and the cognitive walkthrough were selected for the comparison study since they are currently considered the leading inspection methods in industry. Evaluating the UPI with respect to other inspection methods involved a three-part process:

1. Establishing a baseline set of real usability problems from a lab-based usability test of an address book program.
2. Performing independent inspections of the address book program using expert usability practitioners, each assigned to one of three inspection methods: UPI, heuristic evaluation, or cognitive walkthrough.
3. Using the baseline set of usability problems from the lab-based usability test, conducting a comparative analysis of the inspection methods using measures such as thoroughness, validity, effectiveness, severity ratings, and usability.

The intent of the comparison study is to provide useful information regarding the effectiveness of inspection methods for finding real usability problems. In addition, this study seeks to provide the usability practitioner with information that may help them choose between lab-based usability testing and expert-based inspection as the appropriate method to meet their own evaluation objectives.

LAB-BASED USABILITY TEST

The purpose of the lab-based usability test was to generate a set of real usability problems known to impact users interacting with an address book program. Usability problems collected from the lab-based usability test could then be used in a comparative analysis with problems identified from the expert-based inspection methods. The following sections describe the participants, materials, and procedures used to collect the usabilitv problem information.

Method

Participants

Twenty students (18 males, 2 females) enrolled in CS 3724: Introduction to Human-Computer Interaction (Fall, 1999) participated in the lab-based usability test. Students volunteered and received extra credit for their participation in the lab-based usability test. One participant was a graduate student, while the other nineteen were undergraduate students majoring in Computer Science, Computer Engineering, or Industrial and Systems Engineering. As required by the study, none of the participants had any experience with the address book program evaluated in the lab-based usability test.

Materials and Equipment

The InTouch address book program was the target application for this study. InTouch was selected on the basis of its relatively simple interface, corresponding relationship to a paper-based address book, limited fielding in the consumer market, and known usability issues documented in a previous evaluation at Virginia Tech (McCreary, 1996). As an address book program, InTouch approximated most home or office software products within the scope of user experience, interest, and applicability.

The lab-based usability test took place in the Usability Methods Research Lab at Virginia Tech, using a standard usability observation setup shown in Figure 6-1. The InTouch program ran on a Macintosh PowerPC 7500 located in the subject room as shown in Figure 6-2. A Panasonic SVHS videocassette recorder recorded both audio and video and displayed the session on a Sony Triniton monitor shown in Figure 6-3. In addition, a Macintosh computer captured live screen images and mixed with the audio and video so that the videocassette tape would have the information (audio, video, and computer screen image) required for further analysis of usability problems.

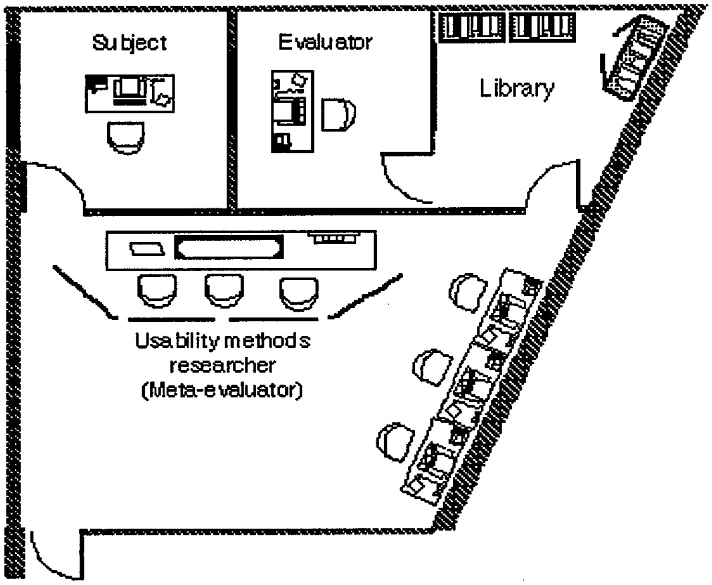


FIGURE 6-1. Observation Setup in the Usability Methods Research Lab at Virginia Tech.

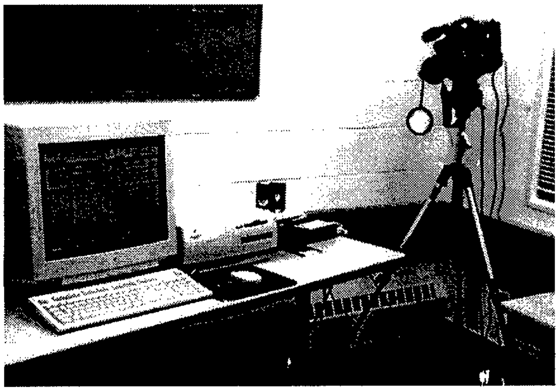


FIGURE 6-2. Participant Setup with Macintosh Computer and Video Camera.

FIGURE 6-3. Observation Room with Videocassette Recorder, Mixer, and Monitor.

Procedure

Each participant began the session with a tour of the Usability Methods Research Laboratory followed by an explanation on how the data collection would occur during the study. Prior to the collection of data, participants read a description of the purpose of the study and signed the Informed Consent Form included in Appendix D. Participants then completed a pre-test questionnaire (Appendix E) and read a description of the InTouch interface and the tasks they would perform during the session (Appendix F). The participants performed the following six tasks:

1. Insert a record.
2. Save a file under a different name.
3. Find a specific record.
4. Sort a file.
5. Make a new group.
6. Import data from a file.

Each participant performed the six tasks using the InTouch program, while verbalizing their thoughts by thinking-aloud during task performance. After completing each task, the participant vocalized any issues not noted during task completion and then took a short break before beginning the next task. The experimenter recorded usability problems based on the participant's performance and verbal protocol during each task. Following completion of all tasks, the participant completed the post-test questionnaire (Appendix G) and was thanked for participating in the study.

Results

Although the primary purpose of the lab-based usability test was to generate a list of real problems for analysis during the comparison study, the results reported here also highlight task timing and completion data as well as user reports in order to provide a comprehensive analysis of the usability test. Analysis of the unique problems is presented first followed by task completion and timing data, and finally user reports.

Usability Problem Identification

Each test participant produced a list of problems that the evaluator observed and recorded. The experimenter recorded a usability problem if one of the following occurred:

1. Participant gave up on task completion and asked the evaluator for help.
2. Participant performed an error that required recovery to proceed with successful task completion.
3. Participant verbalized confusion and/or difficulty when performing the task.
4. Participant visibly showed a delay in accomplishing a part of the task.

Problems that appeared on multiple occasions across participants received the same label and description in order to generate an accurate list of common ar.d unique problems. The experimenter examined video tapes of the recorded sessions if more information was needed on a specific problem following the lab-based test. A total of 39 unique usability problems were identified from the lab-based usability test as shown in Table 6-1. Table 6-1 lists problems by identification number (1-39), relevant task number (1-6), description, interface location, and their frequency of occurrence (out of 20 users). The list is also sorted by frequency of occurrence, from highest to lowest.

TABLE 6-1. Unique Usability Problems Identified During Lab-Based Usability Testing.

Problem id.	Task #	Description	Interface Location	Freq.
1	2	Expected that any actions dealing with the entire address file would be under the File menu (e.g., Save as, Save, Open).	Menubar	19
4	4	Confused by search model because of the labels Last Word and First Word. Expected the use of Last Name or First Name for pick-list labels.	Sort dialog box	19
12	6	Import does not provide prompt for where to put group members. Instead it puts them in the current group.	Import dialog box	19
8	5	Group Search option is not an obvious choice for adding members to a group.	Menubar	17
16	5	The New command under the InTouch menu might lead user to think it is associated with making a new group or new record.	Menubar	12
13	1	No labels provided to help user differentiate between top and bottom entry fields on main screen.	Main screen	11
14	4	Unnecessary prompt to save the file when sorting records, regardless of change status.	Sort dialog box	11
20	3	Find dialog uses "Address" and "Notes" fields but these fields are not identified on the interface.	Find dialog box	11
18	1	Users may not be confident that their new record has been added to the list. No Add or "Do It" button.	Main screen	10
9	6	No global Undo provided. User has to individually delete records that have been erroneously placed in the wrong group.	Main screen	9
11	6	The purpose of the lower left list box is not apparent and the function of the checkmarks is confusing.	Main screen	7
3	3	System provides poor feedback for indicating that the entry has been saved.	Main screen	6
17	6	Poor feedback showing current Group selected.	Main screen	6
19	5	Add button in Group Editing dialog box is not colocated with entry field.	Edit Groups	6
22	1	Insert button does not provide enough information to tell user what it does.	Main screen	6
24	5	Find dialog box has to be dismissed after completing Group Search.	Find dialog box	6
27	5	Edit Groups options does not allow user to manipulate group membership.	Edit Groups	6

TABLE 6-1 (continued)

Problem id.	Task #	Description	Interface Location	Freq.
2	3	User has trouble locating Find command in the menubar. Not located with Edit.	Menubar	5
26	6	Expected that any actions dealing with Importing files would be under the File menu.	Menubar	4
5	4	Results from Sort operation are hard to interpret because of the free-form nature of the address line.	Main screen	3
6	4	Sort dialog box does not go away after the sort is complete. User has to manually dismiss dialog box.	Sort dialog box	3
25	6	User cannot select multiple items to move to another group.	Main screen	3
10	6	Unnecessary confirmation message for deleting individual records.	Main screen	2
15	4	No Cancel button provided for Sort. System only provides Done button.	Sort dialog box	2
23	5	Edit Groups is not an obvious choice for adding or making new groups.	Menubar	2
37	5	New Group button at bottom of Group Search dialog box is difficult to notice.	Group Search dialog box	2
7	5	In Edit Groups, "Done" is not an obvious choice after entering new group.	Edit Groups	1
21	1	Clicking on Insert highlights a blank line in the left window, drawing attention to this window instead of the entry window.	Main screen	1
28	6	Design does not support accessibility. For example, cannot check/uncheck items in Groups View list via the keyboard. Cannot use keyboard Delete key as keyboard shortcut for Delete... function.	Main screen	1
29	4	The difference between the Sort and Done buttons is not clear. Inconsistent with OK and Cancel conventions.	Sort dialog box	1
30	6	Unnecessary extra step required to make new group before Importing.	Menubar	1
31	4	Default value for sort should be Last Word and it should be at top of list.	Sort dialog box	1
32	6	Import does not provide option to look at file before Importing	Import dialog box	1
33	5	The difference between the Search and Done buttons is not clear. Inconsistent with OK and Cancel conventions.	Group Search dialog box	1
34	2	Not sure if the Save command saves the whole rolodex or just the record.	Main screen	1

TABLE 6-1 (continued)

Problem id.	Task #	Description	Interface Location	Freq.
35	5	Feedback after a group search is hard to notice. The Results field is not prominent.	Group Search dialog box	1
36	5	Boxes on left side of main screen are not labeled.	Main screen	1
38	1	Adding a new record cannot be accomplished through menus as expected.	Menubar	1
39	4	System does not provide example of Ascending and Descending order options.	Sort dialog box	1

Based on the 39 unique problems, the mean number of usability problem instances recorded for each user was 11.25, as shown by Table 6-2. Five users encountered as many as thirteen usability problems; whereas, two users encountered as few as eight usability problems. A one-sample t-test for the number of usability problems encountered by each participant showed a significant difference, $\underline{t}(19) = 31.09$, $\underline{p} < .001$, indicating that users are different in terms of the number of problems they experience with the InTouch application.

TABLE 6-2. Mean Number of Usability Problems Identified per User During Lab-Based Usability Testing.

	N	Minimum	Maximum	Mean	SD
Number of Usability Problems	20	8.00	13.00	11.25	1.62

Task Completion and Timing Data

The evaluator recorded tasks as successfully completed if the participant was able to perform the task without help during the session. That is, as long as participants did not experience a task blockage, they were allowed to learn from exploring various paths until they reached the successful end state. By using the above definition for successful completion, participants were still able to experience and comment on usability problems, even when they successfully reached the end goal. Table 6-3 summarizes the completion data for each subject across the six tasks. The data in Table 6-3 can be classified into one of two classes; successful completion or unsuccessful completion. Such data are said to be dichotomous and can be tested using Cochran's (1950) Q-statistic, which has the same form as the chi-square statistic. Testing the variance among the six tasks showed a significant difference in task completion rates using Cochran's Q-statistic, Q(5)= 51.78, p < .001. As designed for the study, task difficulty increased with each successive task.

TABLE 6-3. Completion Data by Subject and Individual Tasks.

	Task						
Subject	**Insert a record**	**Save a file under a different name**	**Find a specific record**	**Sort a file**	**Make a new group**	**Import data from a file**	**% of Tasks Completed by Subject**
1	✔	✔	✔				50.0
2	✔	✔	✔	✔			66.6
3	✔	✔	✔	✔			66.6
4	✔	✔		✔			50.0
5	✔	✔	✔				50.0
6	✔	✔	✔		✔		66.6
7	✔	✔	✔	✔			66.6
8	✔	✔	✔			✔	66.6
9	✔	✔	✔	✔	✔		83.3
10	✔		✔				33.3
11	✔	✔					33.3
12	✔	✔		✔			50.0
13	✔	✔	✔		✔		66.6
14	✔	✔	✔	✔	✔	✔	100.0
15	✔	✔	✔				50.0
16	✔	✔	✔			✔	66.6
17	✔	✔	✔	✔			66.6
18	✔	✔				✔	50.0
19	✔	✔	✔	✔			66.6
20	✔	✔	✔	✔			66.6
% of Subjects Completing each Task	100	95	80	50	20	20	

Table 6-4 presents the results for completion time, showing both mean time for each participant and the mean time for each task. A one-way ANOVA analyzed the completion time data to determine the degree of difference between each task. As shown by Table 6-5, there is a significant difference for completion times across the six tasks, F(5, 95) = 50.3, p < .001. A Greenhouse-Geisser (G-Gp) correction was used to confirm the uncorrected F-test since Mauchly's test of sphericity showed heterogeneity of covariance among the treatment conditions. Results showed that a conservative Greenhouse-Geisser (G-Gp) correction confirmed the significant F-test as shown in Table 6-5. A post-hoc analysis (Table 6-6) using a Bonferonni t-test revealed task completion time did significantly increase from Task 1 to Task 6, p < .01.

TABLE 6-4. Time to Complete Each Task.

	Task						
Subject	**Insert a record**	**Save a file under a different name**	**Find a specific record**	**Sort a file**	**Make a new group**	**Import data from a file**	**Mean Time (sec) by Subject**
1	62	36	75	185	301	243	150.33
2	130	50	57	105	237	278	142.83
3	65	55	65	130	261	248	137.33
4	175	68	115	90	195	235	146.33
5	75	67	105	242	285	323	182.83
6	53	41	65	234	140	169	117.00
7	85	40	93	133	210	285	141.00
8	68	50	63	183	322	186	145.33
9	86	37	52	196	75	237	113.83
10	90	109	50	260	313	305	187.83
11	115	83	182	204	264	200	174.67
12	105	88	148	187	260	240	171.33
13	81	57	110	285	175	283	165.17
14	70	57	29	100	117	117	81.67
15	113	58	82	177	391	229	175.00
16	89	57	90	221	199	107	127.17
17	82	52	85	121	203	123	111.00
18	70	62	93	277	203	170	145.83
19	90	38	32	120	301	146	121.17
20	63	49	64	148	274	177	129.17
Mean Time (sec) for each Task	88.4	57.7	82.8	179.9	236.3	215.1	

TABLE 6-5. ANOVA Summary Table of Completion Time Data.

Source	df	SS	MS	F	p	G-Gp
Between						
Subject	19	85566.16	4503.48			
Within						
Time	5	582995.54	116599.11	50.30	.001	.001
Time x Subject	95	220203.29	2317.93			
Total	119	888764.99				

TABLE 6-6. Bonferroni T-Test Summary of Mean Number of Completion Time Data.

(I) Task Completion	(J) Task Completion	Mean Difference (I-J)	SE
Task 1 Time	Task 2 Time	30.65**	6.31
	Task 3 Time	5.60	8.28
	Task 4 Time	-91.55**	16.85
	Task 5 Time	-147.95**	17.17
	Task 6 Time	-126.70**	14.39
Task 2 Time	Task 3 Time	-25.05	7.43
	Task 4 Time	-122.20**	12.75
	Task 5 Time	-178.60**	16.42
	Task 6 Time	-157.35**	13.73
Task 3 Time	Task 4 Time	-97.15**	13.81
	Task 5 Time	-153.55**	18.23
	Task 6 Time	-132.30**	15.04
Task 4 Time	Task 5 Time	-56.40	21.59
	Task 6 Time	-35.15	17.23
Task 5 Time	Task 6 Time	21.25	19.35

**p < .01

User Reports

A pre-test questionnaire (Appendix E), distributed to all participants, obtained demographic data and assessed participant experience levels with computers and address book programs. Summary data are provided in Table 6-7, showing the minimum, maximum, mean, and standard deviation from questions regarding computer and address book experience. Participants rated questions regarding computer and address book experience on a 4-point ordinal scale (1 [Never] to 4 [Every Day]), while the questions relating to personal computer and Macintosh experience used a 10-point ordinal scale (1 [No Experience] to 10 [Very Experienced]). Results showed all users had at least 3 or more years experience using computers (M = 4.00) and limited experience with electronic address book programs (M = 1.55). Data from Table 6-6 indicated participants had more experience with personal computers (M = 8.85) than Macintosh computers (M = 5.35). Results from a Wilcoxon Signed Ranks test showed a significant difference for personal computer vs. Macintosh experience, Z = 3.96, $p < .01$. Since the InTouch address book program was implemented on a Macintosh platform, it could potentially influence performance, especially those that had considerably more experience with Macintosh applications or address book programs. Calculating the Pearson product moment correlation coefficient showed Macintosh experience was not correlated with either the number of problems experienced ($r = 0.06$, $p > .10$) or the total time to complete the six tasks ($r = -0.26$, $p > .10$). However, address book experience showed negative correlation with the number of problems experienced ($r = -0.51$, $p < .05$) and positive correlation for the total time to complete the six tasks ($r = 0.43$, $p < .10$). Correlations using address book experience indicate that participants with some level of electronic address experience encountered fewer problems, but took longer to complete the tasks.

TABLE 6-7. Summary Data from Lab-Based Usability Test Pre-Test Questionnaire.

	N	Minimum	Maximum	Mean	SD
Computer Experience	20	4.00	4.00	4.00	.000
Address Book Experience	20	1.00	4.00	1.55	.998
PC Experience	20	7.00	10.00	8.85	.988
Mac Experience	20	1.00	7.00	5.35	1.631

After completing the lab-based usability test, participants completed a post-test questionnaire (Appendix G) to assess their ratings of the InTouch interface. Summary data are provided in Table 6-8, showing the minimum, maximum, mean, and standard deviation from questions regarding computer and address book experience. All questions used a Likert-type scale from 1 (Strongly Disagree) to 5 (Strongly Agree) with 3 as the neutral mid-point. Most areas were only slightly above 3.0 (Neutral) indicating participants did not feel strongly either way about these areas. Participants provided slightly stronger ratings for InTouch helping them complete their tasks (M = 3.70) and InTouch providing the needed capabilities (M = 3.70). The most negative comment regarding InTouch providing an appropriate level of feedback (M = 2.60).

TABLE 6-8. Summary Data from Lab-Based Usability Test Post-Test Questionnaire.

	N	Minimum	Maximum	Mean	SD
Simple to Use	20	1.00	5.00	3.35	1.089
Quickly Complete Tasks	20	2.00	5.00	3.70	.864
Efficiently Complete Tasks	20	2.00	5.00	3.45	.825
Appropriate Feedback	20	1.00	5.00	2.60	.940
Easy to Recover from Errors	20	2.00	5.00	3.15	.875
Easy to Use Menus	20	1.00	5.00	3.05	1.050
Easy to Find Information	20	1.00	5.00	3.15	1.182
Like Using InTouch	20	1.00	5.00	3.20	1.151
Has Needed Capabilities	20	1.00	5.00	3.70	.979

EXPERT-BASED INSPECTION COMPARISON STUDY

The primary purpose of the comparison study was to determine the effectiveness of the UPI when compared to two other expert-based inspection methods; the heuristic evaluation and the cognitive walkthrough. The secondary purpose was to develop a standard approach for UEM comparison studies with standards, measures, and criteria to demonstrate the effectiveness of the methods considered in this research. Results from the lab-based usability test (i.e., the problem set) were used in the expert-based inspection comparison study to help demonstrate the effectiveness of methods in terms of identifying real problems. The work reported here is intended to provide researchers and practitioners with a more effective and reliable inspection method and an approach for conducting UEM comparison studies using standardized measures and definitions.

Method for Expert-Based Inspections

Participants

Participants were recruited from industry in order to form a relatively homogenous group of usability professionals with experience in usability testing, interface design, and/or the use of expert-based inspection methods. Researchers (e.g., Desurvire et al., 1991; Desurvire, 1994; Desurvire et al., 1992; Doubleday et al., 1997; Jeffries et al., 1991; Jeffries & Desurvire, 1992) have shown the most proficient inspectors are those with usability experience and formal training in areas such as human factors, computer science, human-computer interaction, and cognitive psychology. Thirty participants (14 males, 16 females), randomly assigned to one of the three expert-based inspection methods (10 for each method), participated in the comparison study. All participants had a minimum of two years experience (M = 8.9 years) in the field supporting activities such as test and evaluation, design, research, or teaching. The participants came from organizations where usability engineering (design, test, or evaluation) was a formal part of their daily experience. All participants possessed at least a bachelor's degree in computer science, human factors, psychology, or industrial engineering. A majority (26 of 30) of the participants possessed an advanced degree (masters or PhD). The average age of the participants was 36 years, ranging from 28 to 50 years.

Materials and Equipment

The same address book program, InTouch, from the lab-based test was used for the comparison study. The relatively simple interface of the InTouch application allowed participants to quickly understand the interface issues without previous experience with this specific address book program. A Macintosh PowerBook 520c hosted the InTouch application in order to transport the application to each usability participant's work site. Materials for the UPI method are the same as those described in Chapter 3. For the comparison study, the UPI method was implemented on an IBM ThinkPad and consisted of a Microsoft Access database, Netscape Communicator, Microsoft Personal Web Server, and a series of Active Server Pages (ASP) that linked HTML pages to the database. Participants in the heuristic evaluation method used materials from Nielsen's (1993) description of the method, while participants in the cognitive walkthrough method used materials described in Wharton et al. (1993).

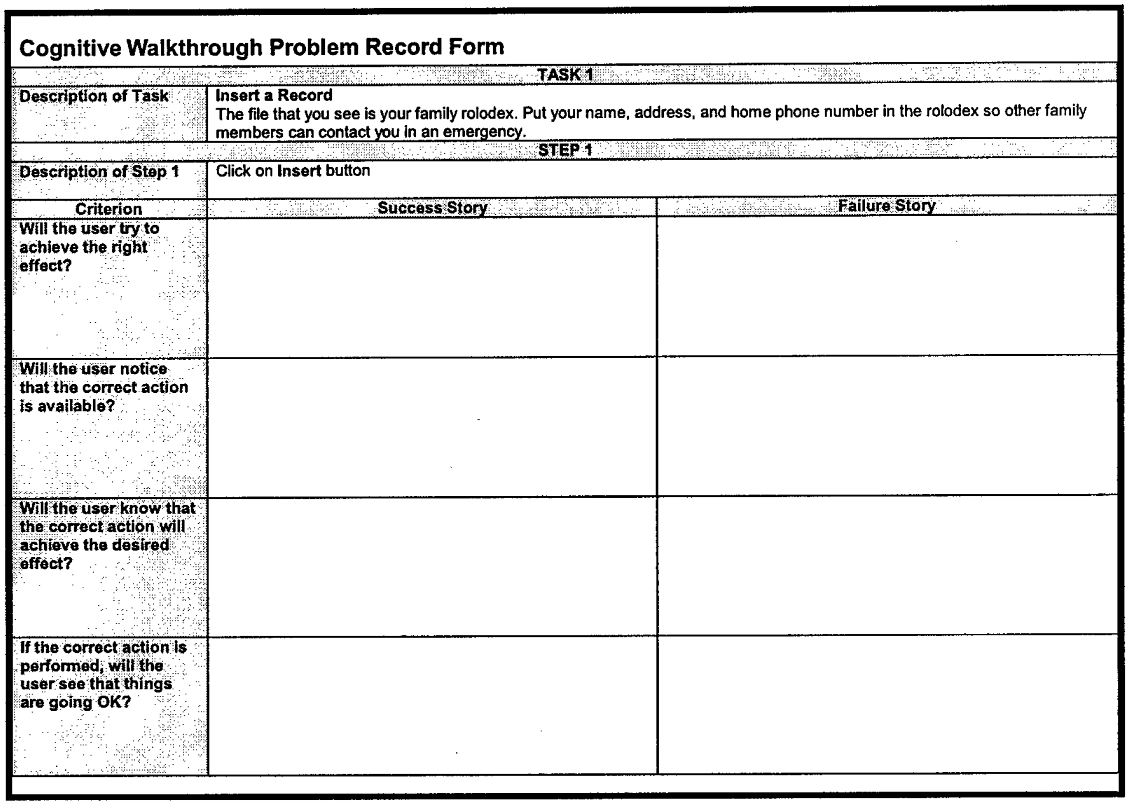
Procedure

The comparative study used a between-groups design with evaluators randomly assigned to one of three groups: UPI, cognitive walkthrough, or heuristic evaluation. To compare the effectiveness of the methods under similar conditions, training and evaluation time was limited to a total of two hours for all methods. The limited training and evaluation time was justified on the basis of the following three reasons. First, all participants were usability practitioners and did not require extensive training, since all had at least a working knowledge of expert-based inspection methods. Second, as volunteers for this study, these usability practitioners had limited time to offer for the evaluation session. Third, the limited evaluation time approximated what is now becoming a typical situation in product development cycles: limited time to complete evaluation activities.

All participants began their evaluation session by completing the Informed Consent Form provided in Appendix H, followed by a pre-test questionnaire (Appendix I) to gather demographic data. Participants then completed a 30-minute training program on their respective method.

Training materials for the UPI method, detailed in Appendix J, consisted of briefing slides explaining both content and structure of the UPI, an information sheet on the InTouch interface, instructions on how to conduct the evaluation, a listing of the six tasks used in the lab-based usability test, and a user-class definition. UPI participants used a web-based tool, detailed in Chapter 3, to document the problems they identified with the interface. UPI participants had one hour to complete their inspection using the six representative tasks from the lab-based usability test. After this task-based inspection, participants had 15 minutes to perform a free-exploration inspection that was not task-dependent. During the free-exploration session, UPI participants could review and inspect any part of the InTouch interface.

The training materials for the cognitive walkthrough participants, detailed in Appendix K, consisted of a tutorial adapted from Wharton et al. (1992), an information sheet on the InTouch interface, instructions on how to conduct the evaluation, a listing of the six tasks used in the lab-based usability test, and a user-class definition. Cognitive walkthrough participants used the paper form shown in Figure 6-4 to document the problems they identified with the interface. Each form included the task description and the step needed to successfully perform the relevant portion of the task. Cognitive walkthrough participants were given 1 hour and 15 minutes to complete their inspection using the six representative tasks from the lab-based usability test. Since the cognitive walkthrough method is only defined for a task-based approach, the inspection did not include a free-exploration portion.

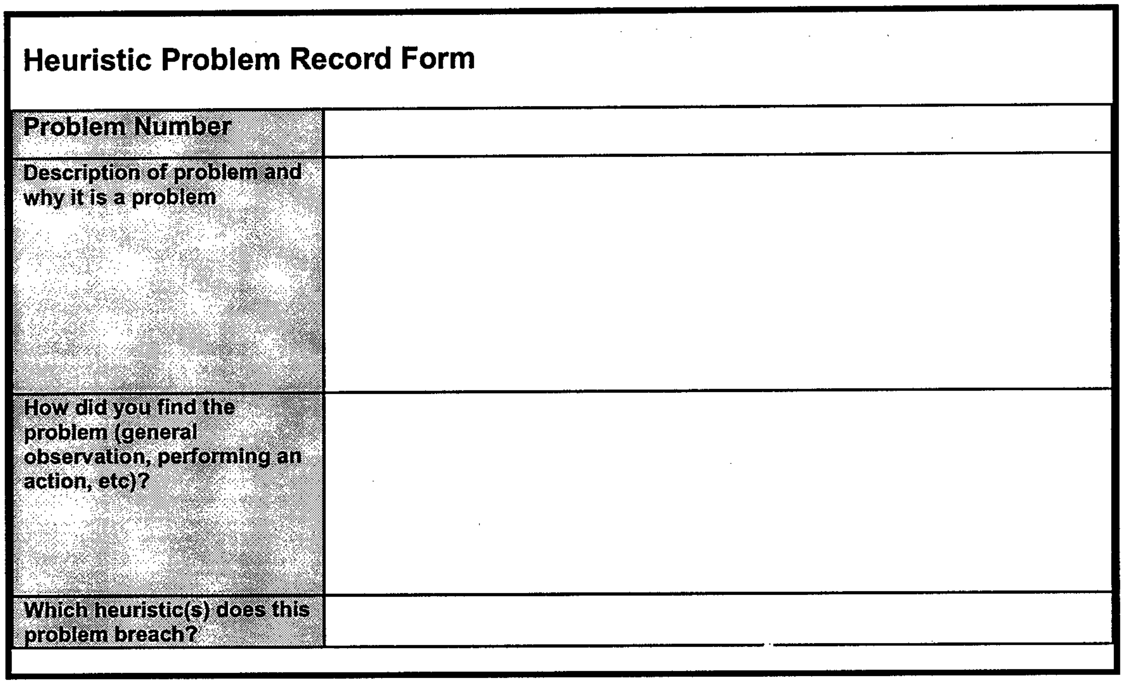


Cognitive Walkthrough Problem Record Form

TASK 1		
Description of Task	Insert a Record The file that you see is your family rolodex. Put your name, address, and home phone number in the rolodex so other family members can contact you in an emergency.	
STEP 1		
Description of Step 1	Click on Insert button	
Criterion	Success Story	Failure Story
Will the user try to achieve the right effect?		
Will the user notice that the correct action is available?		
Will the user know that the correct action will achieve the desired effect?		
If the correct action is performed, will the user see that things are going OK?		

FIGURE 6-4. Problem Documentation Form Used in The Cognitive Walkthrough Method.

The training materials for the heuristic evaluation participants, detailed in Appendix L, consisted of a tutorial adapted from Nielsen (1993), an information sheet on the InTouch interface, instructions on how to conduct the evaluation, a listing of typical tasks performed by users, and a user-class definition. Heuristic evaluation participants used the paper form shown in Figure 6-5 to document the problems they identified with the interface. Each form included a place to record the problem description, how the problem was found, and the heuristic label relevant to the problem. Heuristic evaluation participants were given 1 hour and 15 minutes to complete their inspection using a free-exploration approach. Although heuristic evaluation participants received a list of the typical tasks users perform, they did not follow a task-based approach since the heuristic evaluation, as defined by Nielsen, is designed for a free-exploration inspection approach. Participants were, therefore, free to inspect any part of the interface without the requirement to follow a specific task approach.



Heuristic Problem Record Form

Problem Number	
Description of problem and why it is a problem	
How did you find the problem (general observation, performing an action, etc)?	
Which heuristic(s) does this problem breach?	

FIGURE 6-5. Heuristic Problem Record Form Used to Document Usability Problems.

Following completion of all tasks, each participant completed the post-test questionnaire (Appendix M) and received thanks for participating in the study. The entire inspection session for each method, including training and evaluation, lasted two hours.

Hypotheses

Data from previous UEM studies helped establish potential differences among the three expert-based inspection methods considered in this research. This background data, and the approach taken in the current research leads to five important hypotheses:

1. There will be a significant difference between the three inspection methods in terms of thoroughness, validity, and effectiveness. Support for this hypothesis will indicate that the three methods are indeed different, each offering their own advantages and disadvantages.

2. The UPI and cognitive walkthrough methods will produce significantly higher validity and effectiveness scores than the heuristic evaluation method. Problems

identified within the UPI and cognitive walkthrough methods should more closely resemble problems from lab-based usability testing since both methods use task-based approaches. Thus, the UPI and cognitive walkthrough methods should identify fewer false positives and false negatives than the heuristic evaluation method.

3. The heuristic evaluation method will produce a significantly higher thoroughness score than the UPI and cognitive walkthrough methods. Because the heuristic evaluation method focuses on a free-exploration approach and is relatively easy to use, inspectors will be able to cover much more of the interface than UPI and cognitive walkthrough inspectors. This increased coverage should result in heuristic evaluation inspectors reporting a larger set of problems.

4. The heuristic evaluation method will find significantly more minor usability problems than the UPI and cognitive walkthrough methods. As a result of its sole use of the free-exploration approach, heuristic inspectors will identify issues not necessarily related to a particular task, thus, revealing many more "cosmetic" problems.

5. Cost effectiveness in terms of number of evaluators required should be relatively high for the UPI and heuristic evaluation. That is, the UPI and the heuristic methods should require slightly fewer evaluators than the cognitive walkthrough to achieve a given thoroughness score (e.g., 80% problem detection) because each evaluator can cover much more of the interface with these two methods.

Data Analysis

Each inspection session produced a list of problems for each participant. All problems were tagged with participant and condition identifiers and combined into one list. Each problem fell into one of three categories: (1) the problem was the same as one identified from the lab-based usability test, (2) the problem was common to one or more of the inspection-based methods, or (3) the problem was unique to the method. Duplicate problems identified by the same evaluator were removed from the list. The list of problems was normalized by merging problem descriptions of duplicate problems across conditions into one common description. If a method identified a problem from the lab-based problem set (Table 6-1), the lab-based description of the problem was used, since it had already been well defined before the comparison study. All the analysis discussed in the following sections used normalized problem sets. Herein the normalized problem set is referred to as problem types.

Combining problem types from the lab-based usability test (39) with new problem types identified in the expert-based inspections (62) produced a list of 101 problem types. These 101 problem types formed the basis for the problem severity ratings.

Method for Problem Severity Ratings

Participants

Two human factors graduate students provided severity ratings of the problems identified from the comparison study and the lab-based usability test. The two participants had experience with the InTouch interface, but were not involved in the lab-based usability test or the inspection comparison study. Both participants had equal academic training in human factors and usability engineering.

Procedure

The participants received prepared descriptions of the 101 usability problems in random order. The participants also received information on the InTouch interface and task descriptions from the lab-based usability test. Each participant independently assigned severity ratings using Rubin's (1994) problem severity ranking shown in Table 6-9. After completing their assigned ratings, the two participants produced a combined list of severity ratings by consensus.

TABLE 6-9. Problem Severity Rating Form

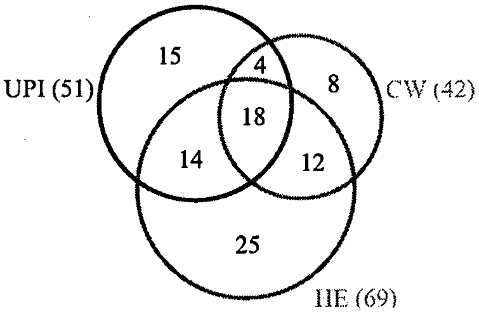
Severity Rating	**Severity Description**	**Severity Definition**
4	Unusable	The user either is not able to or will not want to use a particular part of the product because of the way that the product has been designed and implemented.
3	Severe	The user will probably use or attempt to use the product here, but will be severely limited in his or her ability to do so. The user will have great difficulty in circumventing the problem.
2	Moderate	The user will be able to use the product in most cases, but will have to undertake some moderate effort in getting around the problem.
1	Irritant	The problem occurs only intermittently, can be circumvented easily, or is dependent on a standard that is outside the product's boundaries. Could also be a cosmetic problem.

Source: Rubin (1994)

Results

Problem Types

Participants using the expert-based inspection methods identified a total of 96 problem types. Of these 96 problem types, 62 were new problems while 34 problems came from the original 39 problems already identified from the lab-based usability test. Figure 6-6 shows the problem types divided into seven partitions according to problems unique to each method and common to two or more methods. The area of each circle reflects the total number of problem types identified in each method; actual numbers are indicated in parenthesis. Table 6-10 summarizes several chi-square tests of independence for the number of problem types identified by each method. As shown in Table 6-10, a chi-square test indicated a significant difference between the three inspection methods for the number of problem types identified, $\chi^2(2) = 57.19$, $\underline{p} <.001$. A further test compared the number of problem types between the UPI and heuristic evaluation (HE) methods, and between the cognitive walkthrough (CW) and HE methods. The chi-square test revealed significantly more problem types identified in the HE method (69) as compared to the UPI method (51), $\chi^2(1) = 9.38$, $\underline{p} <.01$. In addition, significantly more problem types appeared in the HE method (69) when compared to the CW method (42), $\chi^2(1) = 9.94$, $\underline{p} <.01$. A test comparing the UPI and CW methods revealed no significant difference in the number of problem types for each method (UPI = 51 vs. CW = 42), $\chi^2(1) = 0.94$, $\underline{p} >.10$.



TOTAL = 96

FIGURE 6-6. Total Problems Types Identified by the Expert-Based Inspection Methods.

TABLE 6-10. Summary of Chi-Square Tests of Independence For Number of Problem Types.

Comparison	χ^2	df
UPI (51) vs. HE (69) vs. CW (42)	57.19***	2
HE (69) vs. UPI (51)	9.38**	1
HE (69) vs. CW (42)	9.94**	1
UPI (51) vs. CW (42)	0.94	1

**p < .01, ***p < .001

Problem Detection

Total problems identified by each participant were the basis for calculating the mean number of usability problem types found in each method. Table 6-11 shows the mean number of usability problem types found by a single evaluator for each of the three inspection methods. Individual inspectors found as few as 7 problems with the UPI and CW methods and as many as 20 problems with the HE method. The mean number of problem types identified by each method was analyzed using a one-way ANOVA to determine the degree of difference between the methods. As shown by Table 6-12, a significant difference existed for the mean number of problem types identified between the three methods, F(2, 27) = 5.78, p < .01. A post-hoc analysis (Table 6-13) using a Bonferonni t-test revealed participants in the HE method (M = 15.00) found significantly more problem types than participants in the UPI method (M = 11.80) and participants in the CW method (M = 11.20).

TABLE 6-11. Mean Number of Problem Types Identified for Each of the Expert-Based Inspection Methods.

Inspection Method	N	Minimum	Maximum	Mean	SD
UPI	10	7.00	13.00	11.80	1.93
HE	10	11.00	20.00	15.00	2.83
CW	10	7.00	16.00	11.20	3.16

TABLE 6-12. ANOVA Summary Table of Mean Number of Problem Types Identified.

Source	df	SS	MS	F	p
Method	2	83.47	41.73	5.78	.008
Subjects / Method	27	195.20	7.23		
Total	29	278.67			

TABLE 6-13. Bonferroni T-Test Summary of Mean Number of Problem Types Identified.

(I) Inspection Method	(J) Inspection Method	Mean Difference (I-J)	SE
HE	UPI	3.20*	1.20
	CW	3.80*	1.20
UPI	CW	.60	1.20

*p < .05

The mean number of problem types was also used to calculate detection rates for each method. Detection rates were calculated using the mean number of problem types for each method and dividing by the total number of problem types (96) identified by all three methods. Mean detection rates ranged from 11.7% (11.2/96) for the CW method, 12.3% (11.8/96) for the UPI method, and 15.6% (15.0/96) for the HE method. Researchers (e.g., Lewis, 1994; Nielsen, 1994b; Virzi, 1990; Virzi, 1992; Wright & Monk, 1991) have shown that the probability formula, $1 - (1 - p)^n$, is a good predictor of the number of aggregate evaluators needed to obtain a desired level of overall detection of usability problems. Figure 6-7 shows the detection probability for each method using the formula $1 - (1 - p)^n$. Based on the data illustrated in Figure 6-7, 10 inspectors are needed in the UPI and CW methods to find at least 70% of the problem types identified across all three methods. Accordingly, 7 inspectors are needed in the HE method to find at least 70% of the problem types.

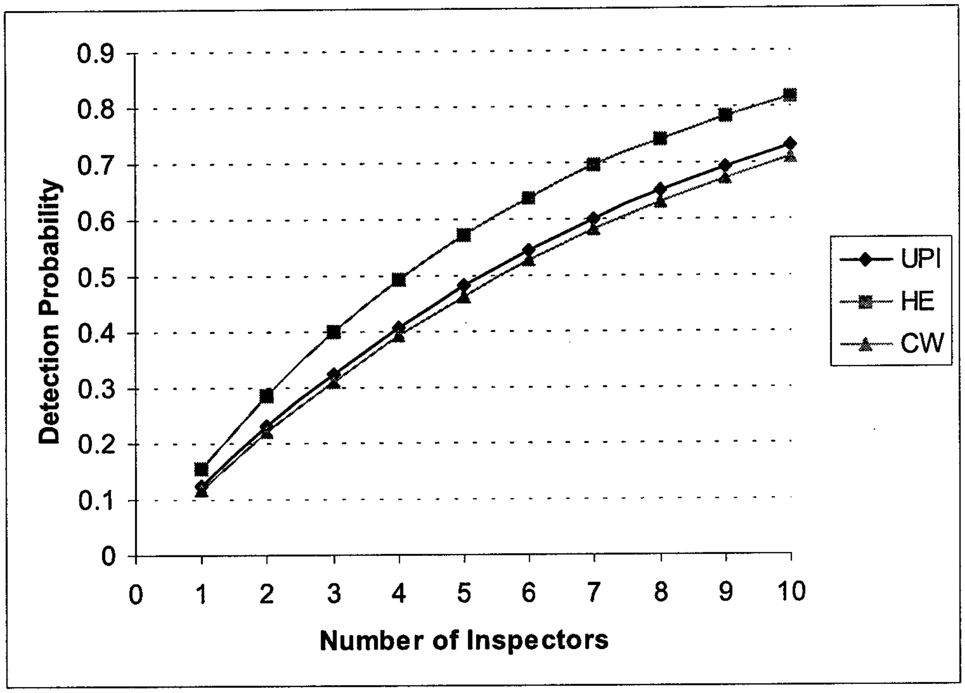


FIGURE 6-7. Detection Probability Based on the Mean Number of Problems Identified by Each Method.

Thoroughness, Validity, and Effectiveness

Thoroughness and validity scores were calculated using the following equations discussed in Chapter 3:

$$Thoroughness = \frac{|\mathrm{P} \cap \mathrm{A}|}{|\mathrm{A}|} = \frac{|\mathrm{P'}|}{|\mathrm{A}|} \tag{2}$$

$$Validity = \frac{|\mathrm{P} \cap \mathrm{A}|}{|\mathrm{P}|} = \frac{|\mathrm{P'}|}{|\mathrm{P}|} \tag{4}$$

where P represents the set of usability problems detected by one of the inspection methods and A represents the set of usability problems identified in the lab-based usability test. Thus, A is defined as the set of real problems that exist in the InTouch interface. Effectiveness is a multiplicative component of thoroughness and validity, defined by the following equation:

$$\text{Effectiveness} = \text{Thoroughness} \times \text{Validity} \tag{11}$$

Results of the calculations for thoroughness, validity, and effectiveness are shown in Table 6-14 and illustrated in Figure 6-8.

TABLE 6-14. Thoroughness, Validity, and Effectiveness of Inspection Methods.

	Inspection Method					
Measure	**UPI**		**HE**		**CW**	
	M	SD	M	SD	M	SD
Thoroughness	.233	.025	.179	.032	.202	.068
Validity	.785	.111	.482	.132	.699	.119
Effectiveness	.182	.028	.089	.036	.146	.065

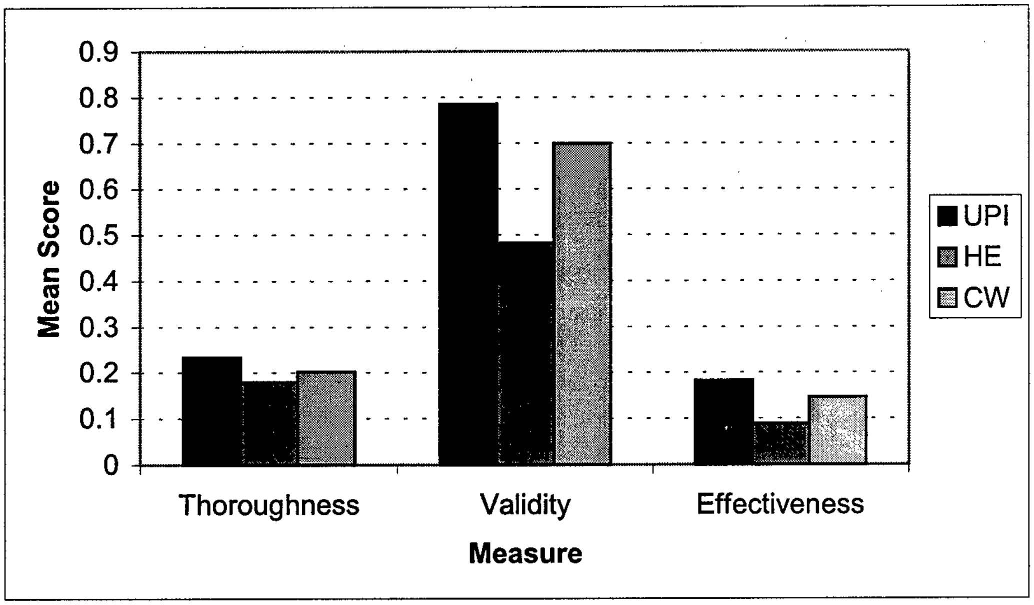


FIGURE 6-8. Mean Value of Thoroughness, Validity, and Effectiveness.

Correlations between the three measures of thoroughness, validity, and effectiveness are shown in the correlation matrix in Table 6-15. As expected, the results showed the measures of thoroughness, validity, and effectiveness to be highly correlated (p < .01), since each of the measures share a common numerator. Based on the significant correlation between thoroughness, validity, and effectiveness, a multivariate analysis of variance (MANOVA) was used to test the difference between the three inspection methods. Results from the MANOVA indicated a significant difference between the three inspection methods using Wilks' Lambda as the test statistic (Lambda = .405), F(6, 50) = 4.74, p < .01. Based on the significance from the MANOVA results, univariate tests determined whether significant differences existed for each measure.

TABLE 6-15. Correlation Matrix for Thoroughness, Validity, and Effectiveness Measures.

	Thoroughness	Validity	Effectiveness
Thoroughness	1.000	.570*	.876*
Validity	.570*	1.000	.881*
Effectiveness	.876*	.881*	1.000

*p < .05

Follow-up one-way ANOVA tests for thoroughness, validity, and effectiveness are summarized in Table 6-16. Results showed a significant difference for thoroughness (F[2, 27] = 3.49, p < .05), validity (F[2, 27] = 16.60, p < .001), and effectiveness (F[2, 27] = 11.94, p < .001). The Bonferonni t-test was used to conduct a post-hoc analysis for multiple comparisons across the three inspection methods. Table 6-17 summarizes the statistics from the Bonferonni t-test. The results of the post-hoc analysis indicate:

- The UPI method (M = .233) was significantly more thorough at identifying problems from the lab-based usability test than the heuristic evaluation method (M = .179), p < .05,

- The heuristic evaluation method (M = .482) is significantly less valid than either the UPI (M = .785) or cognitive walkthrough (M = .699) methods, p < .01,

- The heuristic evaluation method (M = .089) is significantly less effective than either the UPI (mean = .182) or cognitive walkthrough (M = .146) methods, p < .05, and

- The UPI and cognitive walkthrough methods are not significantly different with respect to thoroughness (M_{UPI} = .233 vs. M_{CW} = .202), validity (M_{UPI} = .785 vs. M_{CW} = .699), and effectiveness (M_{UPI} = .182 vs. M_{CW} = .146).

TABLE 6-16. ANOVA Summary Table for Measures Of Thoroughness, Validity, and Effectiveness.

Source	df	SS	MS	F	p
Thoroughness					
Method	2	.015	.007	3.49	.045
Subjects / Method	27	.056	.002		
Total	29	.071			
Validity					
Method	2	.488	.244	16.60	.000
Subjects / Method	27	.397	.014		
Total	29	.885			
Effectiveness					
Method	2	.045	.022	11.94	.000
Subjects / Method	27	.050	.002		
Total	29	.095			

TABLE 6-17. Bonferroni T-Test Summary for Thoroughness, Validity, and Effectiveness.

			Mean	
	(I) Inspection	(J) Inspection	Difference	
Dependent Variable	**Method**	**Method**	**(I-J)**	**SE**
Thoroughness	UPI	HE	.054*	.020
		CW	.031	.020
	HE	CW	-.023	.020
Validity	UPI	HE	.303**	.054
		CW	.086	.054
	HE	CW	-.217**	.054
Effectiveness	UPI	HE	.094**	.019
		CW	.037	.019
	HE	CW	-.057*	.019

*p < .05, **p < .01

Problem Frequency

Data from the lab-based usability test not only provided a real set of problems, but also provided frequency data for each problem. That is, users encountered some problems more frequently than others. Referencing Table 6-1, some problems were experienced by 95% (19/20) of the users while other problems were only experienced by 5% (1/20) of the users. Results from the thoroughness measure were refined using the weighted thoroughness (by frequency) equation discussed in Chapter 3:

$$\textit{Weighted Thoroughness (by frequency)} = \frac{\sum \mathrm{f}\ (\mathrm{rpf}_i)}{\sum \mathrm{f}\ (\mathrm{rpe}_i)} \qquad (18)$$

Using the usability problem data from the lab-based test and substituting frequency counts, instead of simple counts, provided a refinement to the thoroughness scores as shown in Table 6-18.

TABLE 6-18. Comparison of Thoroughness and Weighted Thoroughness Measures.

	Inspection Method		
Measure	**UPI**	**HE**	**CW**
	M	M	M
Thoroughness	.233	.179	.202
Weighted Thoroughness (by frequency)	.444	.221	.359

A one-way ANOVA showed a significant difference for weighted thoroughness when using frequency data from the lab-based usability test, F(2, 27) = 11.7, p < .001. A post-hoc Bonferroni t-test (Table 6-19) revealed the heuristic evaluation method (M = .221) was significantly less thorough at finding frequently occurring problems than the UPI (M = .444), p < .001, or the cognitive walkthrough (M = .359), p < .05. The UPI and the cognitive walkthrough did not show a significant difference in identifying frequently occurring problems from the lab-based usability test (p > .10).

TABLE 6-19. Bonferroni T-Test Summary of Weighted Thoroughness Measure.

(I) Inspection Method	**(J) Inspection Method**	**Mean Difference (I-J)**	**SE**
UPI	HE	.224***	.048
	CW	.086	.048
HE	CW	-.224*	.048

*p < .05, ***p < .001

Problem detection rates were calculated for thoroughness (unweighted and weighted) scores using the mean values from Table 6-18 and the probability formula, $1 - (1 - p)^n$. Figure 6-9 shows the detection probability using the unweighted thoroughness score for each method. Based on the data illustrated in Figure 6-9, 6 inspectors would be needed in the UPI method to detect 80% of the problems identified in the lab-based usability test. The cognitive walkthrough would require 7 inspectors, and the heuristic evaluation method would require 8 inspectors to detect the same level of 80% of the problems from the lab-based usability test. Detection rates improve even more when focusing on only the most frequently occurring problems from the lab-based usability test. Figure 6-10 shows the detection probability using the weighted thoroughness score. As shown in Figure 6-10, only 3 inspectors would be needed in the UPI method to detect 80% of the problems identified in the lab-based usability test. Four inspectors from the cognitive walkthrough method would be needed to detect 80% of the problems from the lab-based usability test. The heuristic evaluation method would require as many as 7 inspectors to detect at least 80% of the problems from the lab-based usability test.

Problem Severity

Severity ratings were collected on all 101 problems from the lab-based usability test and the expert-based inspections. In addition to independent ratings, the two participants produced one list of severity ratings, combining their independent ratings on a consensus basis. Table 6-20 shows the correlation matrix for Rater 1, Rater 2, and Group severity ratings. Severity ratings between Rater 1 and Rater 2 were positively related, $\underline{r} = .432$, $\underline{p} < .01$. Correlations between each rater and the group rating were also positively related, $\underline{r} = .640$ for Rater 1 and Group ($\underline{p} < .01$), and $\underline{r} = .746$ for Rater 2 and Group ($\underline{p} < .01$).

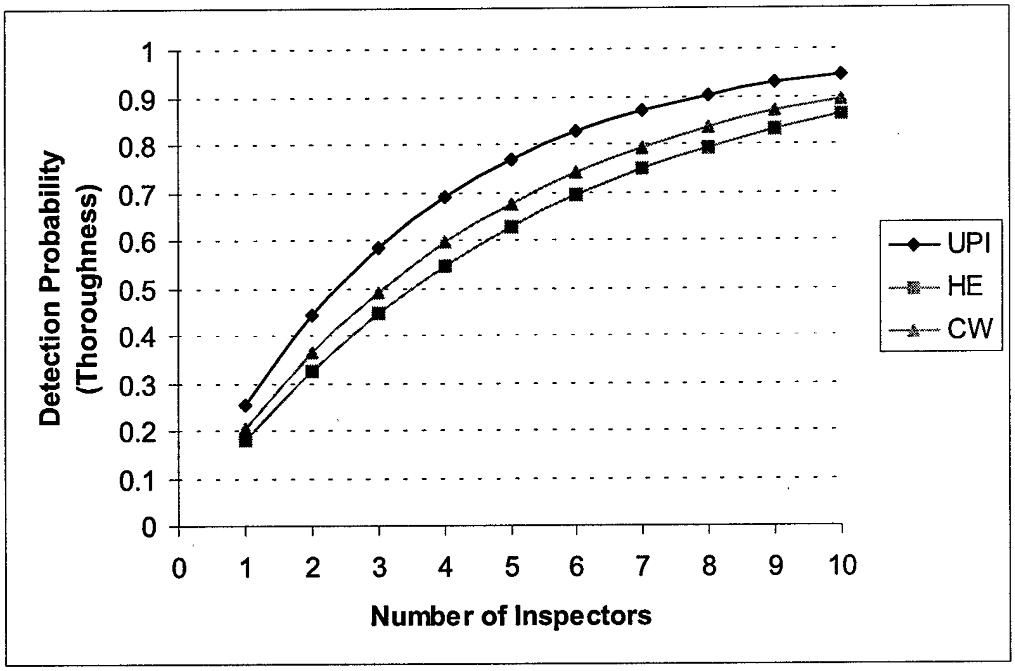


FIGURE 6-9. Detection Probability Based on the Mean Thoroughness Score for Each Method.

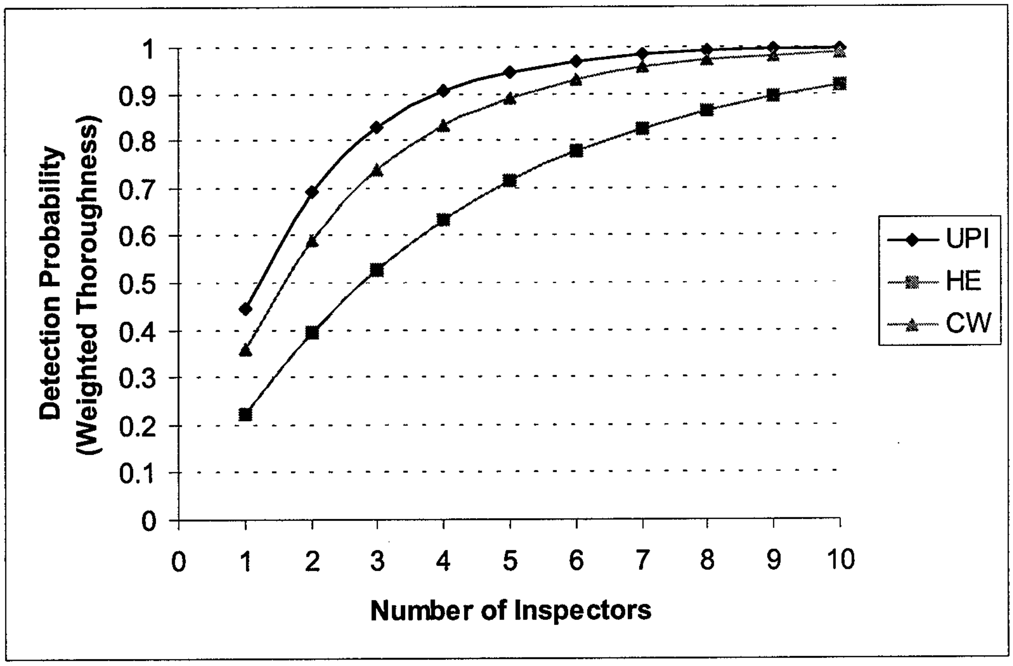


FIGURE 6-10. Detection Probability Based on the Mean Weighted Thoroughness Score for Each Method.

TABLE 6-20. Correlation Matrix for Rater 1, Rater 2, and Group Severity Ratings.

	Rater 1	**Rater 2**	**Group**
Rater 1	1.000	.432**	.640**
Rater 2	.432**	1.000	.746**
Group	.640**	.746**	1.000

**p < .01

Kendall's coefficient of concordance assessed the overall degree of agreement among the participants' ratings of severity. A high degree of concordance was obtained, W(100) = .716, p < .05, indicating that the two raters viewed the problems at comparable levels of severity. Based on the positive agreement found among raters, the group severity ratings were used to conduct further analyses related to usability problem severity.

The mean severity ratings for each of the inspection methods are shown in Table 6-21. Mean ratings ranged from 2.16 for the heuristic evaluation to 2.21 and 2.22 for the cognitive walkthrough and UPI, respectively. A one-way ANOVA (Table 6-22) showed no significant difference between the mean severity ratings, F(2, 27) = .488, p > .10. Thus, no clear distinction can be made among the three inspection methods using problem severity ratings. According to Rubin's (1994) problem severity ranking, the mean ratings reported for all three inspection methods (M = 2.16 to 2.22) is only slightly above a rating of *Moderate* (reference Table 6-9). According to Rubin, a Moderate rating means "the user will be able to use the product in most cases, but will have to undertake some moderate effort in getting around the problem" (p. 279).

TABLE 6-21. Mean Severity Ratings for Each of the Inspection Methods.

Inspection Method	**N**	**Mean**	**SD**	**Minimum**	**Maximum**
UPI	10	2.22	.109	2.08	2.38
HE	10	2.16	.185	1.77	2.33
CW	10	2.21	.147	1.89	2.38

TABLE 6-22. ANOVA Summary Table of Mean Severity Ratings.

Source	df	SS	MS	F	p
Method	2	.022	.011	.488	.619
Subjects / Method	27	.611	.023		
Total	29	.633			

A further test was determined the difference between severity ratings of lab-based problems compared to the unique problems only identified through one of the inspection methods. Severity raters did not receive information that identified the source of the problem (i.e., whether the problem came from the lab test or from the expert-based inspections). Thus, this further analysis tests whether severity ratings were applied differentially to problem types originating from the lab-based usability test as compared to unique problem types identified in the expert-based inspection methods. Table 6-23 presents the mean severity ratings listed by originating source of the problem. The expert-based inspection methods also identified 34 of the 39 problems from the lab-based usability test. However, for this analysis, only the unique problems are credited to the inspection-based methods in order to test the difference between real (lab-based) and non-real (inspection-based unique) problem types. A one-way ANOVA test for severity of real (M = 2.10) vs. non-real problems (2.23) did not reveal a significant difference, F(1, 99) = .988, p > .10.

TABLE 6-23. Mean Severity Ratings Identified by Source of Problem.

Source of Problem	N	Mean	SD	Minimum	Maximum
Real (lab-based)	39	2.10	.640	1.00	3.00
Non-real (inspection-based unique)	62	2.23	.584	1.00	4.00

User Reports

All participants completed a pre-test questionnaire (Appendix I) in order to obtain demographic data and assess overall experience in the field and experience levels with usability evaluation activities. Overall usability experience used an interval scale with total months as the scale value. Questions regarding usability evaluation experience (usability testing, interface design, cognitive walkthrough, and heuristic evaluation) used the following 6-point ordinal scale:

1: Never
2: About once per year
3: Few times per month
4: Few times per month
5: Few times per week
6: Daily

Summary data are provided in Table 6-24, showing the minimum, maximum, mean, and standard deviation from questions regarding overall usability experience and experience with usability testing, interface design, the cognitive walkthrough, and the heuristic evaluation method. Results showed all users had a minimum of 2 years of usability ($\underline{M}$ = 8.97). Results also showed that participants had the most experience in interface design (mean = 4.03) and the least experience with the cognitive walkthrough technique ($\underline{M}$ = 2.20). Experience with usability testing (mean = 3.20) and the heuristic evaluation technique ($\underline{M}$ = 3.17) fell roughly in the middle. A Friedman test revealed participants had significantly different experience levels in terms of these four usability evaluation activities, $\chi^2(3) = 39.59$, $\underline{p} < .001$. Table 6-25 summarizes the results from a post-hoc Wilcoxon Signed Ranks analysis, showing interface design experience significantly greater than experience in usability testing ($\underline{p} < .01$), heuristic evaluation ($\underline{p} < .01$), and cognitive walkthrough ($\underline{p} < .001$). In addition, experience with the cognitive walkthrough was significantly less than experience with usability testing ($\underline{p} < .01$) and heuristic evaluation ($\underline{p} < .01$). Although these results showed experience varied among participants, this variance was not significant when distributed among the three inspection methods, $\underline{F}(2, 27)$ = .296 to .760, $\underline{p} > .10$. Therefore, random assignment of participants to one of the three inspection methods distributed experience levels relatively evenly.

TABLE 6-24. Summary Data from Expert-Based Inspection Method Pre-Test Questionnaire.

	N	Minimum	Maximum	Mean	SD
Overall Usability Experience	30	2.00	18.00	8.97	4.70
Usability Testing Experience	30	2.00	5.00	3.20	.96
Interface Design Experience	30	1.00	6.00	4.03	1.43
CW Experience	30	1.00	5.00	2.20	1.29
HE Experience	30	1.00	5.00	3.17	1.12

TABLE 6-25. Summary of Wilcoxon Signed Ranks Analysis of Usability Experience Levels.

Comparison	Z
Interface Design > Usability Testing Experience	2.72**
Interface Design > Heuristic Evaluation Experience	2.61**
Interface Design > Cognitive Walkthrough Experience	4.43***
Usability Testing > Cognitive Walkthrough Experience	2.82**
Heuristic Evaluation > Cognitive Walkthrough Experience	3.29**

**p < .01, ***p < .001

After completing the lab-based usability test, participants completed a post-test questionnaire (Appendix M) to assess attributes of their respective inspection method. Summary data is provided in Table 6-26, showing the mean and standard deviation from questions addressing five areas of the inspection method each participant used. All questions used a Likert-type scale from 1 (Strongly Disagree) to 5 (Strongly Agree) with 3 as the neutral mid-point. A MANOVA was used to test the difference between the three inspection methods in terms of the five questions on the post-test. Results from the MANOVA did not reveal a significant difference between the three inspection methods, Wilks' Lambda = .578, F(10, 46) = 1.45, p > .10.

Figure 6-11 displays the results of the post-test questionnaire graphically, indicating general trends with respect to participant ratings. For example, participants rated the UPI (M = 3.90) as somewhat more effective than either the heuristic evaluation (M = 3.50) or the cognitive walkthrough (M = 3.70). In terms of quickly evaluating the interface, the UPI (M = 2.80) faired less favorably than either the heuristic evaluation (M = 3.30) or the cognitive walkthrough (M =

3.20). The cognitive walkthrough scored the highest in terms of learnability (M = 4.00) as compared to the UPI (M = 3.60) and the heuristic evaluation (M = 3.40). In addition, the cognitive walkthrough was seen as somewhat easy to apply (M = 3.40) compared to the UPI (M = 3.20) and the heuristic evaluation (M = 2.90). Finally, participants in the heuristic evaluation group provided a more favorable response in terms of recommending the technique (M = 3.70) as compared to participants in the UPI (M = 3.20) and cognitive walkthrough (M = 3.30) groups.

TABLE 6-26. Summary Data from Expert-Based Inspection Method Post-Test Questionnaire.

	Inspection Method					
Measure	**UPI**		**HE**		**CW**	
	M	SD	M	SD	M	SD
Effective Technique	3.90	.57	3.50	1.27	3.70	1.34
Quickly Evaluate	2.80	.79	3.30	1.06	3.20	1.32
Easy to Learn	3.60	.84	3.40	1.35	4.00	.82
Easy to Apply	3.20	.92	2.90	1.20	3.40	1.51
Recommend Technique	3.20	.79	3.70	1.16	3.30	1.34

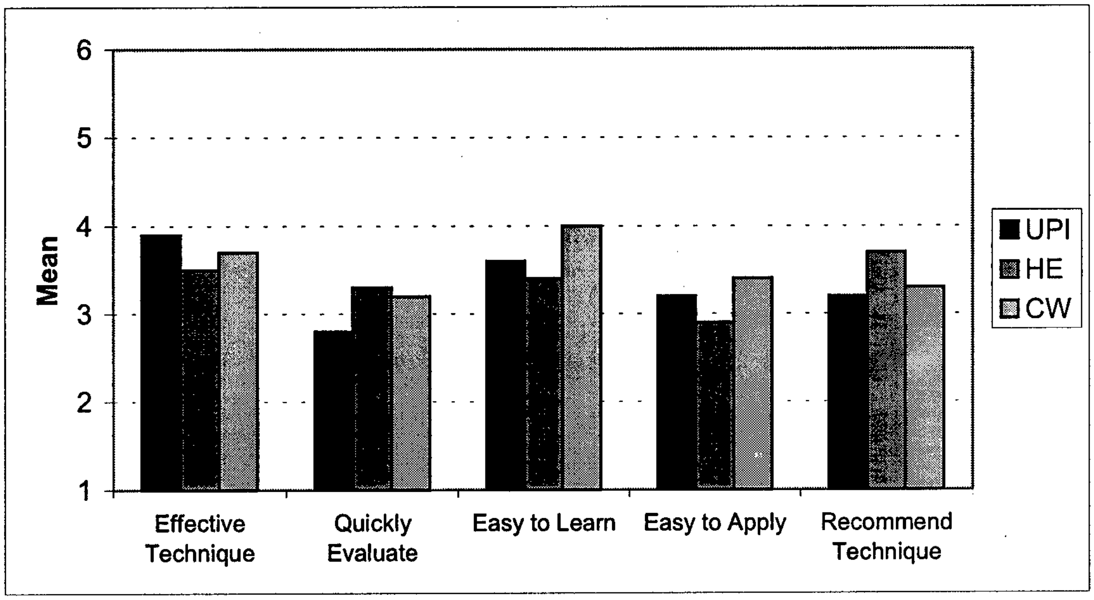


FIGURE 6-11. Mean Response by Inspection Method.

Participants also provided comments on their respective evaluation technique. Table 6-27 presents a summary of the top three positive and negative comments provided by participants in each group. The UPI technique was generally seen as a very organized approach, but will require extensive usage before a user will feel comfortable with the entire framework. Flexibility and an open-ended evaluation were strong attributes of the heuristic evaluation. However, participants pointed out the difficulty they had when assigning a specific heuristic to a usability problem. Participants viewed the cognitive walkthrough as a technique that provided a methodological approach to inspecting an interface, but could be very time consuming if lots of tasks needed evaluation.

TABLE 6-27. Summary of Positive and Negative Comments for Each Inspection Method.

Inspection Method	**Positive Comments**	**Negative Comments**
UPI	• The UPI helped to organize my review. • The UPI helped me think about potential problems that at first I could only vaguely describe. • Problem reporting mechanism is very useful. It helps to see previous problem reports.	• The UPI needs an easier/quicker way to navigate between nodes in the framework. • Need capability to visualize the entire framework of questions; what is coming next and what issues are adjacent. • Need considerable usage before developing a full understanding of the entire knowledge base.
Heuristic Evaluation	• Provides lots of flexibility for performing an evaluation to match my style. • Heuristics provide good reminders of issues to look for and are relatively easy to remember. • Good method to use when I have limited time to perform an evaluation.	• Difficult to assign a heuristic to certain problems, especially when they had multiple issues. • No structure provided to help conduct the evaluation. • The names of the heuristics do not exactly match their respective descriptions.
Cognitive Walkthrough	• Provides a consistent way to evaluate an interface for each task and step. • Method was helpful for making me think about potential issues along the way. • Helped me evaluate interface from user's perspective.	• Time consuming if there are a lot of tasks to evaluate. • Hard to separate out issues related to the four questions. Many of the questions overlap and are redundant. • Does not allow evaluator to explore alternative paths that users may take.

DISCUSSION

The purpose of performing a lab-based usability test and combining with an expert-based inspection comparison study was to determine if a theory-based framework and tool could be effectively used to find important usability problems in an interface design. Overall results from this study did show the UPI to be an effective tool for usability inspection. In addition, data from both the lab-based usability test and the expert-based inspection comparison study point to several findings that are important considerations for conducting UEM comparison studies. Considerations for both the lab-based usability study and the expert-based inspection study are discussed in the following sections.

Lab-Based Usability Test

The lab-based usability test generated a baseline set of real usability problems known to impact users. Through asymptotic user testing with 20 participants, a total of 39 problem types were documented with the InTouch interface. The assumption was that 20 participants were needed in order to approximate the full set of problems likely to exist in the interface and that adding any more participants would not contribute greatly to the total problem set. This assumption arose from data by Lewis (1994) and Virzi (1992) indicating that average problem detection rates can range from 0.16 to 0.42 for any one individual. The mean detection rate for problems identified in the lab-based usability test was 0.29 (mean problem identification of 11.25 divided by 39 problems). As it turns out, a smaller number of users could have been used to generate an asymptotic list of usability problems. Using the formula $1 - (1 - p)^n$ with $p = 0.29$, Figure 6-12 shows how the problem discovery likelihood rises with each added user up to 20 users. Thus, for the data from the lab-based usability test, the probability formula $1 - (1 - p)^n$ predicts as few as 13 users would be needed to find 99% of the problems existing in the InTouch application. However, an increase in the number of participants would have been warranted had the participants, on average, experienced much less than 29% of the problems. Extending the usual lab-based usability test to include several more users increased the confidence of generating a realistic and complete set of real problems in the InTouch application that could then be used as part of the actual criterion for the comparison study.

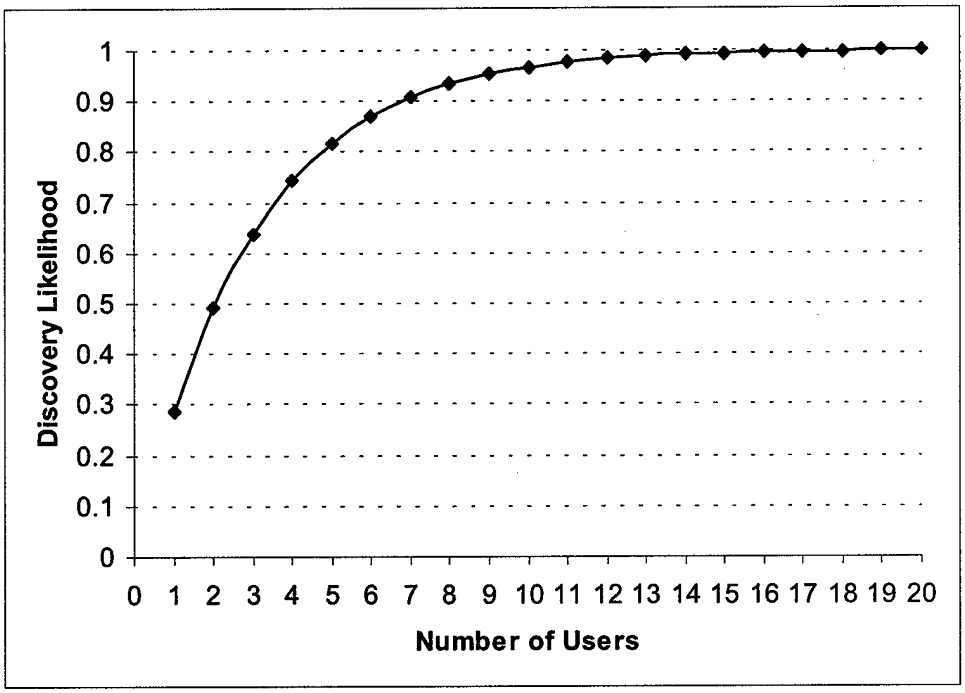


FIGURE 6-12. Problem Discovery Likelihood Based on an Individual Detection Rate of 0.28.

Expert-Based Inspection Comparison Study

The baseline set of real problems from the lab-based usability test formed the key component of the criteria for comparing expert-based inspection methods. Using a structured framework of usability concepts and issues, the UPI proved to be quite effective in finding important usability problems from an address book program. Since the analysis focused on one particular interface application, the results are limited to similar home and office applications that do not require extensive training or years of experience. Although other interface applications were available for analysis, the InTouch address book program simulated many of the available applications that home and business users currently use. In addition, expert-based inspection methods are primarily directed at "walk-up-and-use" products. UEM comparison study research has not shown the type of interface style to impact usability problem identification, other than the fidelity of the prototype as reported by Nielsen (1990b). Table 6-28 presents a summary of the research hypotheses and their corresponding result.

TABLE 6-28. Summary of Research Hypotheses Results.

Hypothesis	Result
1. There will be a significant difference between the three inspection methods in terms of thoroughness, validity, and effectiveness. Support for this hypothesis will indicate that the three methods are indeed different, each offering their own advantages and disadvantages.	Supported: A significant difference was found for the measures of thoroughness, validity, and effectiveness.
2. The UPI and cognitive walkthrough methods will produce significantly higher validity and effectiveness scores than the heuristic evaluation method. Problems identified within the UPI and cognitive walkthrough methods should more closely resemble problems from lab-based usability testing since both methods use task-based approaches. Thus, the UPI and cognitive walkthrough methods should identify fewer false positives and false negatives than the heuristic evaluation method.	Supported: The heuristic evaluation method was significantly less valid and effective than the UPI and cognitive walkthrough. The UPI and cognitive walkthrough were found to be comparable.
3. The heuristic evaluation method will produce a significantly higher thoroughness score than the UPI and cognitive walkthrough methods. Because the heuristic evaluation method focuses on a free-exploration approach and is relatively easy to use, inspectors will be able to cover much more of the interface than UPI and cognitive walkthrough inspectors. This increased coverage should result in heuristic evaluation inspectors reporting a larger set of problems.	Not supported: The heuristic evaluation detected more raw problems, but many of these were false alarms, reducing its overall thoroughness score.
4. The heuristic evaluation method will find significantly more minor usability problems than the UPI and cognitive walkthrough methods. As a result of its sole use of the free-exploration approach, heuristic inspectors will identify issues that are not necessarily related to a particular task, thus revealing many more "cosmetic" problems.	Not supported: Severity ratings did not produce a significant difference among the three methods.
5. Cost effectiveness in terms of number of evaluators required should be relatively high for the UPI and heuristic evaluation. That is, the UPI and the heuristic methods should require slightly fewer evaluators than the cognitive walkthrough to achieve a given thoroughness score (e.g., 80% problem detection) because each evaluator can cover much more of the interface with these two methods.	Not supported: Because of the great number of false alarms identified in the heuristic evaluation method, it requires significantly more evaluators to obtain a given thoroughness level.

Problem Detection

Problem detection rates among the three inspection methc ds were relatively low when compared to other similar research involving expert evaluators. For example, Nielsen (1994b) has reported an average detection rate of 30% for any one evaluator. Mack and Montaniz (1994) have reported average detection rates varying from 36% to 45%. For this study, detection rates ranged from 11.7% for the cognitive walkthrough, 12.3% for the UPI, and 15.6% for the heuristic evaluation. These rates are much lower than those reported by Nielsen, Mack and Montaniz, and other researchers. However, data from Lewis (1994) indicate that detection rates can be as low as 16% for certain software programs, especially office applications. Thus, it is likely that the InTouch interface is replete with a great number of usability problems that are difficult for any one evaluator to find much more than a small percentage. Another consideration is the limited amount of time each evaluator was given to inspect the InTouch interface. The procedure for the expert-based inspection study controlled for evaluation time, limiting each evaluation session to a total of 1 hour and 15 minutes. Had the evaluation been extended for several hours, it is very likely that detection rates would be in the range reported by other similar evaluation studies. Although the participants had limited evaluation time, Dumas et al. (1995) have pointed out in practical application it is more effective to have a greater number of evaluators examine an interface for a shorter period of time than just a few evaluators who spend a large amount of time. The result of using several evaluators for short periods of time inspecting an interface generally produces better coverage of detecting important usability problems.

Problem detection rates were not given significant consideration in developing important criteria for usability methods because merely finding a large number of problems is more about productivity than it is effectiveness. Detection rates are essentially about raw production, without any penalty for identifying false alarms. A method that helps an evaluator find hundreds of problems will most likely score well in terms of problem detection, yet could be very ineffective for finding real problems that exist. In addition, a method that finds a large number of problems presents a huge cost to the developer who has to process all the problem reports, applying some filtering mechanism to determine the important problems. Therefore, measures such as thoroughness and validity were included to assess the more important attributes of the expert-based inspection methods.

Thoroughness, Validity, and Effectiveness

Results from calculating thoroughness, validity, and effectiveness provided better data in terms of differentiating each of the expert-based inspection methods. Although the heuristic method showed a considerable advantage for identifying the greatest number of problems, it showed significantly less performance when its problem set was compared with the problem set from the lab-based usability test. The heuristic evaluation method was found to be significantly less thorough than the UPI method. Thoroughness is very similar to the detection rate measure, except that thoroughness filters out all the false positives that each evaluator identifies. That is, if an evaluator identifies a problem not from the lab-based usability test, then that problem is not included in the total problem count. The fact that the heuristic method showed decreased performance when using the thoroughness measure highlights its propensity to find a large number of false positives. Other researchers (e.g., Jeffries et al., 1991; Sears, 1997) have confirmed the conclusion that the heuristic evaluation method finds a large number of "non-real" problems.

Similar results were obtained for both the validity and effectiveness measures. That is, the heuristic evaluation method was less valid and, therefore, less effective than either the UPI or cognitive walkthrough methods. Validity measures how much extra effort is being spent on issues that are not important. As an example, the heuristic evaluation method produced a validity score of 0.482. Even though 48.2% of the problems identified by any one inspector using the heuristic evaluation method were valid, 51.8% of the effort was wasted in finding problems that were not part of the real set. Contrast the heuristic evaluation result with results from the UPI and the cognitive walkthrough. Inspectors using the UPI identified 78.5% valid problems on average, while only wasting 21.5% of their efforts on problems that turned out to be not related to the lab-based usability problem set. The cognitive walkthrough inspectors identified 69.9% valid problems on average, while spending 30.1% of the time identifyirg problems outside of the lab-based usability problem set.

The effectiveness measure combines both thoroughness and validity into a figure of merit score. Such a measure accounts for the fact that a method scoring high in thoroughness could conceivably generate a large number of invalid problems. Because the heuristic evaluation method scored low in both thoroughness and validity, its effectiveness score was also the lowest for effectiveness.

One important finding is that the UPI and cognitive walkthrough were relatively equivalent for all three measures. Although the UPI had a slight advantage in raw number form for all three measures, its advantage was not significant. Both the UPI and the cognitive walkthrough implemented task-based approaches to inspecting the InTouch application, while the heuristic evaluation used a free-exploration approach. Sears (1997) also used task-based inspection methods and found them to be much more valid than a free-exploration method (i.e., heuristic evaluation). However, in one case, Sears found a task-based approach (cognitive walkthrough) to be less thorough than a free-exploration approach (heuristic evaluation). In the present study, the cognitive walkthrough and the heuristic evaluation method were roughly equivalent in terms of thoroughness. Task-based approaches generally take more time than free-exploration, but expose the inspector to issues that are most likely going to impact real users. Thus, free-exploration methods tend to identify a large number of false alarms or miss a good percentage of important problems.

Including frequency data in the thoroughness calculation provided a more refined view of the differences between the three methods. Frequency data are often reported in UEM comparison studies (e.g., Cuomo, 1993; Nielsen, 1994b; Virzi, 1990), but without a corresponding relationship to a thoroughness measure. In addition, frequency data are typically reported on problem counts from inspectors, not from users in a lab-based usability test. Frequency data obtained from the inspectors across the various methods in a study does not necessarily imply importance. It could just imply that the problem is very obvious to the inspectors and, therefore, everyone notices it (even though, for example, it's not all that important). However, frequency data from users in a lab-based usability test indicate that certain problems are more likely to surface when the system is fully deployed in the field. Frequency data from the users in the lab-based usability test were subsequently included as part of the thoroughness measure, producing a weighted thoroughness measure. The weighted thoroughness measure reported in the present study provides a more accurate reflection of the types of problems identified by each method. Again, the heuristic evaluation method was significantly less thorough than either the UPI or the cognitive walkthrough. Higher weighted thoroughness scores for both the UPI and cognitive walkthrough methods indicate they are able to find a greater percentage of the frequently occurring problems. Such a result is potentially beneficial to a practitioner who wants to use a method to find the largest number of problems that users will frequently encounter.

Results from the thoroughness and weighted thoroughness measures provided valuable information in terms of the number of inspectors required for each method. Inspection methods are typically implemented by aggregating the results from two or more inspectors. Lewis (1994) and Virzi (1992) have shown the probability formula, $1 - (1 - p)^n$, to be an excellent predictor of aggregated results from groups of evaluators. The UPI curve showed the steepest increase in detection probability for both thoroughness and weighted thoroughness. This result means that the UPI requires fewer inspectors to find a given proportion of existing problems than either the cognitive walkthrough or heuristic evaluation. The heuristic evaluation method presented the highest cost in terms of number of inspectors required to find a given proportion of usability problems. Because of its free-exploration approach, the heuristic evaluation method not only identifies a great number of false alarms, it also requires many more inspectors to obtain a comparable level of thoroughness seen in the UPI or cognitive walkthrough.

Problem Severity

Problem severity ratings did not produce the expected differentiation among the three inspection methods. Although good agreement existed between the two raters, the eventual group severity ratings were somewhat conservative and appeared to center around a mean rating only slightly higher than 2.0. Thus, the majority of problems were of moderate severity according to Rubin's (1994) problem severity ranking scale. Two issues can potentially explain the reason for a lack of differentiation among severity ratings. First, the raters may have been somewhat conservative in giving out severe ratings. Since the raters did not actually see the problems reported, they may have had difficulty in noting which problems deserved the most severe ratings. Second, some important information may have been lost in the normalization process for determining the problem descriptions. Equivalent problems identified by different methods were described using a combination of the two descriptions, with a goal of capturing the essence of the problem. The researcher accomplished this process in the most objective way possible, but the process is still inherently subjective. In either case, raters witnessed a list that has built-in information loss that potentially impacts their ratings of the problems. Attributes that make some problems more severe than others may have been lost in the process. Likewise, certain characteristics that make a problem seem less severe might have been missing in the final list of problem descriptions.

An additional consideration is the rating scale descriptions used for reporting severity information. Research involving UEM studies has not produced a consistent definition of severity. For example, Nielsen (1994b) uses a five-point scale for rating severity; with 0 representing "not a usability problem" and 4 representing a "usability catastrophe." Rubin's (1994) scale, used in this research, includes 4 points; with 1 representing an "irritant" and 4 representing "unusable." Other researchers (e.g., Desurvire et al., 1992; Desurvire & Thomas, 1993; Dutt et al., 1994; Karat et al., 1992) have used their own definitions of severity as part of the study. Research by Andre, Williges, and Hartson (1999) has shown that no single UEM consistently identifies the most severe problems. In the final analysis, severity ratings do not appear to be as useful as measures such as thoroughness and validity.

User Reports

Results from the post-test questionnaire did not show significant differences between the three methods, but did reveal some interesting subjective opinions. The UPI was slightly more effective than both the heuristic evaluation and cognitive walkthrough. However, participants using the UPI did not view it as a method they could use to perform a quick evaluation. This opinion is most likely due to the enormity of the question database. Inspectors may find it difficult to realize that parts of the database are quickly pruned with a "no" answer to certain questions. In addition, the current version of the UPI involves a careful traversal process that has several checks and balances to ensure the user is tracking to the desired point. Users can quickly learn certain parts of the database and desire to return quickly to a node they have visited before, but the method still requires them to traverse the structure.

One of the more interesting ratings was the cognitive walkthrough was seen as easy to learn and apply by the participants using this method. This finding is contradictory to previous research on the cognitive walkthrough where it has been viewed as the most difficult method to learn and apply (Lewis et al., 1990; Rowley & Rhoades, 1992; Wharton et al., 1992). However, some of the research on the cognitive walkthrough has been conducted with undergraduate and graduate students who may not have had the necessary experience to fully understand the cognitive aspects involved with the cognitive walkthrough method. Participants in the present research had considerable experience in the field and appropriate academic training that may have been compatible with the requirements of the cognitive walkthrough.

Another interesting observation is that the heuristic evaluation method was not rated highly effective, yet participants would recommend the technique to their organization. This result could be due to the popularity of the heuristic evaluation, and participants would find it conflicting to not recommend the technique, when in fact, many use this technique in their own usability evaluation activities.

CHAPTER 7. CONCLUSION

This research set out to accomplish two primary goals. First, to develop a usability inspection method that is effective in finding important usability problems in an interface design. Second, to develop a UEM comparison approach with standards, measures, and criteria to prove the effectiveness of methods.

A major part of the research effort involved the development of the UAF. The UAF is a theory-based, interaction-style-independent structured knowledge base of usability issues and concepts. The UAF is based, as an adaptation and extension of Norman's (1986) theory of action model, on what a user perceives and does throughout each cycle of interaction with a machine. The UAF is an essentially hierarchical structure of usability attributes. Users (usability problem classifiers) traverse the UAF as a decision structure, selecting the most appropriate classification category and sub-category at each level of the hierarchy. The cumulative set of category choices along the classification path is taken as a sequence of usability attributes that determines a complete classification description of the usability problem in question. An important goal for the UAF was to design in a model and structure of usability concepts and issues that usability professionals could use in a consistent manner. A reliability study was conducted to test the agreement of evaluators classifying a given set of usability problems using the UAF. Results showed that users were in strong agreement when classifying a set of 15 usability problems using the UAF. The agreement was much stronger than results obtained from the heuristic evaluation method and previous work from a taxonomy similar to the UAF. Strong reliability allowed for confident development of an inspection tool to help inspectors find usability problems in an interface design using the same knowledge base from the UAF.

Based on the strong reliability of the UAF, a mapping was made to an inspection tool, the UPI. The goal of the UPI is to help inspectors conduct a highly focused inspection of a target application resulting in a list of usability problems that users will potentially have with the application. The UPI brings together aspects of both the heuristic evaluation and the cognitive walkthrough. While the heuristic evaluation focuses on ease of use, the cognitive walkthrough focuses on completeness and structure. The UPI fits in between these two, capturing the ease of use but also providing interaction-based structure.

The comparison study examined various measures of effectiveness between the UPI, heuristic evaluation, and cognitive walkthrough. A lab-based usability test helped generate a baseline set of real problems known to impact users. The lab-based usability test generated a list of 39 problem types, many of which were also found with one or more of the expert-based inspection methods. Thoroughness, validity, and effectiveness were calculated using the problem set from the lab-based usability test. Results showed the UPI to be significantly more thorough, valid, and effective than the heuristic evaluation method. In addition, the UPI slightly outperformed the cognitive walkthrough in terms of these same three measures, but this performance increase was not significant. Both the cognitive walkthrough and the UPI implemented task-based approaches during inspection, explaining why the two were relatively similar in terms of finding important problems from the lab-based problem set.

Frequency data from the users in the lab-based usability test were included to refine the thoroughness measure into a weighted thoroughness measure. Weighted thoroughness provides an indication of the ability of a method to not only find problems from the lab-based usability test, but to also find the frequently occurring problems; those problems that will impact the greatest number of users. Again, the UPI and the cognitive walkthrough outperformed the heuristic evaluation method in terms of weighted thoroughness. Weighted thoroughness led to the conclusion that fewer evaluators were needed with the UPI and cognitive walkthrough to detect a given proportion of the problems from a lab-based usability test. As few as 3 UPI inspectors would be needed to detect 80% of the problems identified in the lab-based usability test. Four inspectors from the cognitive walkthrough would be needed to detect the same proportion. However, the heuristic evaluation method would require as many as 7 inspectors to detect at least 80% of the problems from the lab-based usability test.

Severity ratings were also collected after generating the entire set of usability problem types from the lab-based usability test and the three inspection methods. Results from the severity ratings did not help to differentiate the three inspection methods. Severity raters essentially gave only slightly higher than a moderate rating for most usability problems. Although previous research has differentiated methods using severity ratings, the results have often been contradictory. In any case, the severity rating approach used in this research could not distinguish the inspection methods on the basis of severe versus minor problems. Thus, severity ratings did not prove to be as useful as thoroughness, validity, and effectiveness.

Table 7-1 summarizes the performance dimensions of the three inspection methods using the measures calculated in this research. Subjective ratings of low, medium, and high were given to each method based on their quantitative performance in the comparison study. In addition to total problems detected, thoroughness, validity, and effectiveness, a fifth added measure summarized the cost effectiveness of each method. Cost effectiveness is about the number of evaluators required to obtain a performance level based on thoroughness. That is, a method is high in cost effectiveness when it requires only a few evaluators to identify the majority of problems from a standard set of usability problems. As shown in Table 7-1, the major strength of the heuristic evaluation is the total number of problems detected in an interface. However, because many of the problems detected by the heuristic method are false alarms, it has low ratings for all other measures of thoroughness, validity, effectiveness, and cost effectiveness. The major strengths of the UPI are in the areas of validity and cost effectiveness. Because the UPI uses both a task-based and free-exploration approach, it is able to find a significant amount of the important problems in an interface design without a huge cost in terms of the number of evaluators required. The cognitive walkthrough tracks very closely with the UPI, since it too is task-based. The cognitive walkthrough does not detect a great number of problems because of its systematic process, but it is able to find the important problems much like the UPI. A disadvantage to the cognitive walkthrough is the lack of a consistent reporting mechanism to help users provide accurate problem descriptions. Although the cognitive walkthrough and the UPI track closely in many of the measures, the UPI offers a potential advantage in helping provide a more complete and consistent description of each identified usability problem.

Figure 7-1 illustrates graphically the performance measures obtained from the expert-based inspection method. The size of the circles in Figure 7-1 represent performance in terms of total problems detected. In addition, the overlap of each circle represents the degree to which any two methods are related in terms of similar performance. Although not a perfect representation of the results, Figure 7-1 illustrates how the UPI and cognitive walkthrough are similar, and these two methods are quite different than the heuristic evaluation. In addition, Figure 7-1 shows how the UPI closely approaches the upper right corner where optimum performance exists.

TABLE 7-1. Summary of Performance of the Expert-Based Inspection Methods.

	Inspection Method		
Criterion	UPI	Cognitive Walkthrough	Heuristic Evaluation
Total Problems Detected	Medium	Low-Medium	High
Thoroughness	Medium	Medium	Low
Validity	High	Medium-High	Low
Effectiveness	Medium	Medium	Low
Cost Effectiveness	High	Medium-High	Low

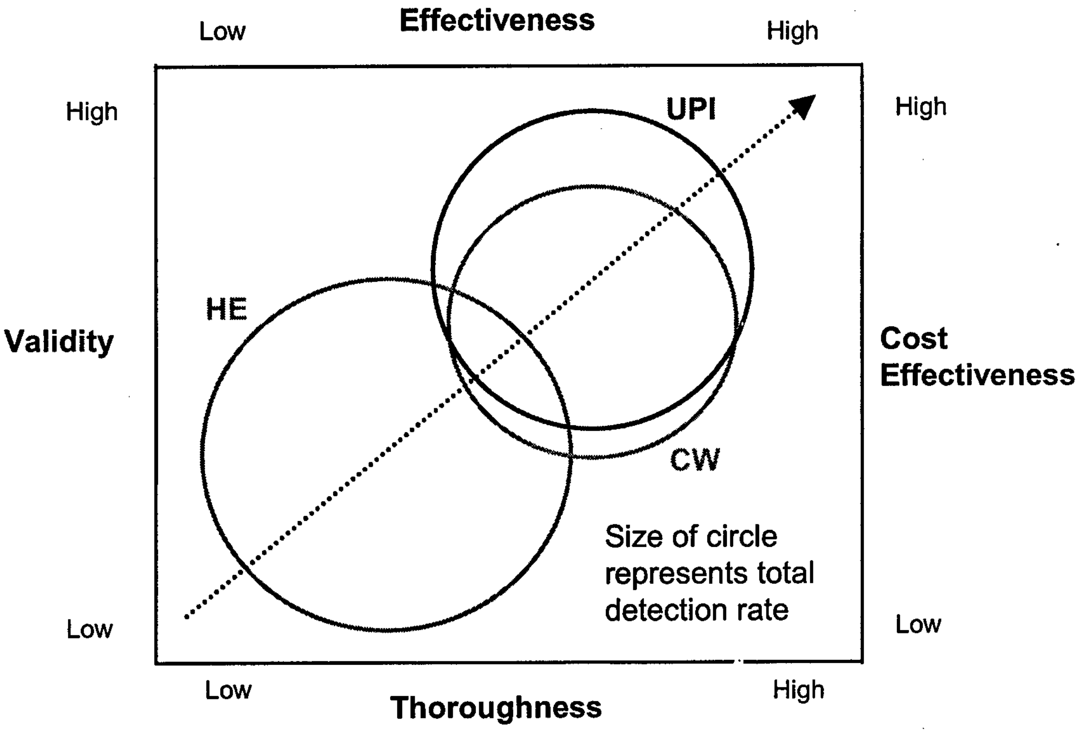


FIGURE 7-1. Venn Diagram of Inspection Method Performance.

Several usability evaluation methods are available to the practitioner for finding usability problems, but none combine a theoretical framework with a usable inspection process. The research reported here provides a unifying framework as a foundation for a structured method and integrated tool to get the most value from usability data by way of usability inspection. How to identify effective evaluation methods in HCI has not been clearly defined and continues to need more research in the development of baseline studies. The research reported in this work attempts to provide some standard measures and methods for consistent and clear comparisons of usability evaluation methods. The hope is that researchers will take note of the candidate measures suggested in this work, as well as the framework provided for usability inspections, and continue to improve the measures, methods, and tools in HCI. Although the UPI is expected to perform more effectively than some other expert-based inspection methods, it is not expected to replace usability testing. The goal is for the UPI to offer an alternative to developers concerned about the time and training demands of the cognitive walkthrough or the overlap and missing areas of the heuristic evaluation.

RESEARCH CONTRIBUTION AND IMPLICATIONS

Most UEMs in the HCI field do not currently employ a method based on a theoretical foundation. As a result, practitioners applying these methods generally find them to be inconsistent and questionable in terms of finding important usability problems without a significant cost in resources. A recent study by Gray and Salzman (1998) documented specific concerns with UEM comparison studies. A key concern noted by Gray and Salzman is the issue of using the right measure (or measures) to compare UEMs in terms of effectiveness. The research presented here answers some of the questions raised by Gray and Salzman. First, standard metrics and criteria can be developed and measured reliably in order to compare UEMs. Thoroughness, validity, and effectiveness appear to form the core of criterion measures researchers should investigate in UEM comparison studies. Thoroughness and validity measures must take into account the question of usability problem realness. Lab-based testing with users appears to be an effective way to provide a "standard" set of real usability problems. Although not an exact replication of real work contexts, user-based lab testing does provide a good indication of the types of problems that actually impact users. Second, investing the time in developing a model-based framework is useful and effective for improving the performance of UEMs. Usability inspection methods without a theoretical foundation experience inconsistent application and often, obtain ineffective results. Researchers and practitioners can benefit from methods and tools designed to support UEMs by facilitating usability problem classification, analysis, reporting, and documentation, as well as usability problem data management (Hartson et al., 1999). In the context of UEM evaluation, a reliable usability framework is as essential component for developing both methods and tools to support usability evaluation activities. The UAF, along with its mapping to an inspection tool, presents an initial model that researchers and practitioners can use to further improve the current methods in usability evaluation.

RECOMMENDATIONS FOR FUTURE RESEARCH

Development of the UPI Tool

The UPI tool used in this research provided a simple way to interact with the knowledge base from the UAF. In most cases, users were able to successfully traverse the UAF hierarchy and reach an appropriate end-node to fully describe a usability problem. However, almost all users noted that the structured traversal of UPI questions was oftentimes more meticulous than necessary, especially after gaining experience with a few usability problems. Users quickly became familiar with the flow of usability questions in the UPI and needed a faster way to a known usability concept. Although the UPI appeared to provide the beginning user with the necessary information to make informed decisions at each question node, this same information slowed down the process for users that quickly understood the structure of the usability concepts and issues. Thus, the UPI tool needs both novice and expert traversal mechanisms to accommodate the various skills and understanding of users. Future versions of the UPI tool might explore the benefits of using a search tool that presents all UPI questions related to a specific usability concept the user wants to examine. The UPI tool could also incorporate a graphical "fast-page" to help both sets of users visualize the hierarchical structure and quickly go to a specific question in the database.

Utility of Usability Problem Reports

The research reported here used a comparison approach based on the raw usability problem lists generated from UEMs. The assumption is that these lists can be compared and measured using several different attributes of effective problem reporting. Future research may be beneficial in determining the actual utility of the list of problems in a way similar to the approach taken by John and Marks (1997). That is, how are problem reports rated in terms of quality or usefulness by developers that have to decipher and process the reports? Methods like the UAF that provide a hierarchical structure and process for describing problems in a more complete way should produce better problem reports.

Determining Problem Severity

Another important research area concerns severity ratings. In the current study, severity ratings did not help differentiate the inspection-based. Andre, Williges, and Hartson (1999) have shown that severity ratings do not consistently identify a single UEM that finds the most severe problems. It is possible that the current measures and definitions of severity are not an accurate characteristic of effective UEMs. More research is needed to develop standard methods and metrics to measure usability problem severity.

Normalization of Usability Problem Lists

A final research area worth investigating is the normalization process often used to combine usability problem lists from two or more inspection methods. Currently, this process is highly subjective, often dependent on researchers and their ability to tell one problem from another. A more automated approach is needed where problems can be separated or combined based on a set of criteria that is validated before the study.

REFERENCES

Andre, T. S., Belz, S. M., McCreary, F. A., & Hartson, H. R. (in press). Testing a framework for reliable classification of usability problems. In Proceedings of the Human Factors and Ergonomics Society 44th Annual Meeting . Santa Monica, CA: Human Factors and Ergonomics Society.

Andre, T. S., Williges, R. C., & Hartson, H. R. (1999). The effectiveness of usability evaluation methods: Determining the appropriate criteria. In Proceedings of the Human Factors and Ergonomics Society 43rd Annual Meeting (pp. 1090-1094). Santa Monica, CA: Human Factors and Ergonomics Society.

Baecker, R. M., Grudin, J., Buxton, W. A. S., & Greenberg, S. (1995). Chapter 2. Design and evaluation. In R. M. Baecker, J. Grudin, W. A. S. Buxton, & S. Greenberg (Eds.), Readings in human-computer interaction: toward the year 2000 (2nd ed., pp. 73-91). San Francisco, CA: Morgan Kaufman Publishers, Inc.

Bailey, R. W. (1972). Testing manual procedures in computer-based business information systems. In Proceedings of the Human Factors Society 16th Annual Meeting (pp. 709-714). Santa Monica, CA: Human Factors Society.

Bailey, R. W. (1997). Usability studies and testing. In T. S. Andre & A. W. Schopper (Eds.), Human factors engineering in system design (pp. 285-308). Wright-Patterson Air Force Base, OH: CSERIAC.

Bastien, J. M., Scapin, D. L., & Leulier, C. (1996). Looking for usability problems with the ergonomic criteria and with the ISO 9241-10 dialogue principles. In CHI '96 Conference Proceedings (pp. 77-78). New York: ACM Press.

Bastien, J. M. C., & Scapin, D. L. (1995). Evaluating a user interface with ergonomic criteria. International Journal of Human-Computer Interaction, 7(2), 105-121.

Beer, T., Anodenko, T., & Sears, A. (1997). A pair of techniques for effective interface evaluation: Cognitive walkthroughs and think-aloud evaluations. In Proceedings of the Human Factors and Ergonomics Society 41st Annual Meeting (pp. 380-384). Santa Monica, CA: Human Factors & Ergonomics Society.

Bennett, J. (1984). Managing to meet usability requirements: Establishing and meeting software development goals. In J. Bennett, D. Case, J. Sandelin, & M. Smith (Eds.), Visual display terminals (pp. 161-184). Englewood Cliffs, NJ: Prentice-Hall.

Bias, R. (1991). Walkthroughs: Efficient collaborative testing. IEEE Software, 8(5), 94-95.

Bias, R. G. (1994). The pluralistic usability walkthrough: Coordinated empathies. In J. Nielsen & R. L. Mack (Eds.), Usability inspection methods (pp. 63-76). New York: John Wiley & Sons.

Blatt, L. A., & Knutson, J. F. (1994). Interface design guidance systems. In J. Nielsen & R. L. Mack (Eds.), Usability inspection methods (pp. 351-384). New York: John Wiley & Sons.

Bradford, J. S. (1994). Evaluating high-level design: Synergistic use of inspection and usability methods for evaluating early software designs. In J. Nielsen & R. L. Mack (Eds.), Usability inspection methods (pp. 235-253). New York: John Wiley & Sons.

Brooks, P. (1994). Adding value to usability testing. In J. Nielsen & R. L. Mack (Eds.), Usability inspection methods (pp. 255-271). New York: John Wiley & Sons.

Butler, K. A. (1996, Jan.). Usability engineering turns 10. Interactions, 3, 58-75.

Card, S. K., Moran, T. P., & Newell, A. (1983). The psychology of human-computer interaction. Hillsdale, NJ: Lawrence Erlbaum Associates.

Carroll, J. M., Kellogg, W. A., & Rosson, M. B. (1991). The task-artifact cycle. In J. M. Carroll (Ed.), Designing interaction: Psychology at the human-computer interface (pp. 74-102). Cambridge, UK: Cambridge University Press.

Cochran, W. G. (1950). The comparison of percentages in matched samples. Biometrika, 37, 256-266.

Cockton, G., & Lavery, D. (1999). A framework for usability problem extraction. In Proceedings of the IFIP Seventh International Conference on Human-Computer Interaction - INTERACT '99 (pp. 344-352). London: IOS Press.

Cohen, J. (1960). A coefficient of agreement for nominal scales. Educational and Psychological Measurement, 20(1), 37-46.

Cuomo, D. L. (1993). A methodology and encoding scheme for evaluating the usability of graphical, direct manipulation style interfaces. In Proceedings of the Human Factors and Ergonomics Society 37th Annual Meeting (pp. 1137-1141). Santa Monica, CA: Human Factors and Ergonomics Society.

Cuomo, D. L. (1994). A method for assessing the usability of graphical, direct-manipulation style interfaces. International Journal of Human-Computer Interaction, 6(3), 275-297.

Cuomo, D. L., & Bowen, C. D. (1992). Stages of user activity model as a basis for user-system interface evaluations. In Proceedings of the Human Factors Society 36th Annual Meeting (pp. 1254-1258). Santa Monica, CA: Human Factors and Ergonomics Society.

Cuomo, D. L., & Bowen, C. D. (1994). Understanding usability issues addressed by three user-system interface evaluation techniques. Interacting with Computers, 6(1), 86-108.

Desurvire, H., Lawrance, D., & Atwood, M. (1991). Empiricism versus judgment: Comparing user interface evaluation methods on a new telephone-based interface. ACM SIGCHI Bulletin, 23(4), 58-59.

Desurvire, H. W. (1994). Faster, Cheaper! Are usability inspection methods as effective as empirical testing? In J. Nielsen & R. L. Mack (Eds.), Usability inspection methods (pp. 173-202). New York: John Wiley & Sons.

Desurvire, H. W., Kondziela, J. M., & Atwood, M. E. (1992). What is gained and lost when using evaluation methods other than empirical testing. In A. Monk, D. Diaper, & M. D. Harrison (Eds.), People and computers VII (pp. 89-102). Cambridge, UK: Cambridge University Press.

Desurvire, H. W., & Thomas, J. C. (1993). Enhancing the performance of interface evaluators. In Proceedings of the Human Factors and Ergonomics Society 37th Annual Meeting (pp. 1132-1136). Santa Monica, CA: Human Factors and Ergonomics Society.

Dix, A., Abowd, G., & Beale, R. (1993). Human-computer interaction. Englewood Cliffs, NJ: Prentice Hall.

Doubleday, A., Ryan, M., Springett, M., & Sutcliffe, A. (1997). A comparison of usability techniques for evaluating design. In Designing Interactive Systems (DIS '97) Conference Proceedings (pp. 101-110). New York: ACM Press.

Draper, S. W. (1986). Display managers as the basis for user-machine communication. In D. A. Norman & S. W. Draper (Eds.), User centered system design: New perspectives on human-computer interaction (pp. 339-352). Hillsdale, NJ: Lawrence Erlbaum Associates.

Dumas, J., Sorce, J., & Virzi, R. (1995). Expert reviews: How many experts is enough? In Proceedings of the Human Factors and Ergonomics Society 39th Annual Meeting (pp. 228-232). Santa Monica, CA: Human Factors and Ergonomics Society.

Dutt, A., Johnson, H., & Johnson, P. (1994). Evaluating evaluation methods. In G. Cockton, S. W. Draper, & G. R. S. Weir (Eds.), People and computers IX (pp. 109-121). Cambridge, UK: Cambridge University Press.

Egan, J. P. (1975). Signal detection theory and ROC analysis. New York: Academic Press.

Fleiss, J. L. (1971). Measuring nominal scale agreement among many raters. Psychological Bulletin, 76(5), 378-382.

Freedman, D., & Weinberg, G. (1990). Handbook of walkthroughs, inspections, and technical Reviews : Evaluating programs, projects, and products. New York: Dorset House.

Goldstein, I. L. (1993). Training in organizations: Needs assessment, development, and evaluation. (3rd ed.). Pacific Grove, CA: Brooks/Cole.

Gould, J. D. (1988). How to design usable systems. In M. G. Helander (Ed.), Handbook of human-computer interaction (pp. 757-789). Amsterdam: Elsevier Science.

Gray, W. D., & Salzman, M. C. (1998). Damaged merchandise? A review of experiments that compare usability evaluation methods. Human-Computer Interaction, 13(3), 203-261.

Hartson, H. R. (1998). Human-computer interaction: Interdisciplinary roots and trends. The Journal of Systems and Software, 43, 103-118.

Hartson, H. R., Andre, T. S., Williges, R. W., & Van Rens, L. (1999). The user action framework: A theory-based foundation for inspection and classification of usability problems. In H. Bullinger & J. Ziegler (Eds.), Human-computer interaction: Ergonomics and user interfaces (Vol. 1, pp. 1058-1062). Mahway, NJ: Lawrence Erlbaum.

Hartson, H. R., & Castillo, J. C. (1998). Remote evaluation for post-deployment usability improvement. In Proceedings of AVI'98 (Advanced Visual Interfaces) (pp. 22-29)ACM.

Hartson, H. R., Castillo, J. C., Kelso, J., Kamler, J., & Neale, W. C. (1996). Remote evaluation: The network as an extension of the usability laboratory. In CHI '96 Conference Proceedings (pp. 228-235). New York: ACM Press.

Hix, D., & Hartson, H. R. (1993). Developing user interfaces: Ensuring usability through product and process. New York: John Wiley & Sons.

Holleran, P. A. (1991). A methodological note on pitfalls in usability testing. Behaviour and Information Technology, 10(5), 345-357.

Jeffries, R., Miller, J. R., Wharton, C., & Uyeda, K. M. (1991). User interface evaluation in the real world: A comparison of four techniques. In CHI '91 Conference Proceedings (pp. 119-124). New York: ACM Press.

Jeffries, R. J., & Desurvire, H. W. (1992). Usability testing vs. heuristic evaluation: Was there a contest? ACM SIGCHI Bulletin, 24(4), 39-41.

John, B. E., & Marks, S. J. (1997). Tracking the effectiveness of usability evaluation methods. Behaviour and Information Technology, 16, 188-202.

John, B. E., & Mashyna, M. M. (1997). Evaluating a multimedia authoring tool. Journal of the American Society for Information Science, 48(11), 1004-1022.

Kahn, M. J., & Prail, A. (1994). Formal usability inspections. In J. Nielsen & R. L. Mack (Eds.), Usability inspection methods (pp. 141-171). New York: John Wiley & Sons.

Karat, C. (1994). A comparison of user interface evaluation methods. In J. Nielsen & R. L. Mack (Eds.), Usability inspection methods (pp. 203-233). New York: John Wiley & Sons.

Karat, C. (1997a). Chapter 32. Cost-justifying usability engineering in the software life cycle. In M. G. Helander, T. K. Landauer, & P. V. Prabhu (Eds.), Handbook of human-computer interaction (pp. 767-778). Amsterdam: Elsevier Science.

Karat, C., Campbell, R. L., & Fiegel, T. (1992). Comparison of empirical testing and walkthrough methods in user interface evaluation. In CHI '92 Conference Proceedings (pp. 397-404). New York: ACM Press.

Karat, J. (1997b). User-centered software evaluation methodologies. In M. G. Helander, T. K. Landauer, & P. V. Prabhu (Eds.), Handbook of human-computer interaction (pp. 689-704). Amsterdam: Elsevier Science.

Karat, J., & Bennett, J. (1991). Working within the design process: Supporting effective and efficient design. In J. M. Carroll (Ed.), Designing interaction: Psychology at the human-computer interface (pp. 269-285). Cambridge, UK: Cambridge University Press.

Keenan, S. L. (1996). Product usability and process improvement based on usability problem classification. Unpublished doctoral dissertation, Virginia Polytechnic Institute and State University, Blacksburg, VA.

Keenan, S. L., Hartson, H. R., Kafura, D. G., & Schulman, R. S. (1999). The Usability Problem Taxonomy: A framework for classification and analysis. Empirical Software Engineering, 4(1), 71-104.

Kieras, D. E., & Polson, P. G. (1985). An approach to the formal analysis of user complexity. International Journal of Man-Machine Studies, 22, 365-394.

Kies, J. K., Williges, R. C., & Rosson, M. B. (1998). Coordinating computer-supported cooperative work: A review of research issues and strategies. Journal of the American Society for Information Science, 49(9), 776-779.

Kurosu, M., Matsuura, S., & Sugizaki, M. (1997). Categorical inspection method-structured heuristic evaluation (sHEM). In 1997 IEEE International Conference on Systems, Man, and Cybernetics (pp. 2613-2618). Piscataway, NJ: IEEE.

Landauer, T. K. (1995). The trouble with computers: Usefulness, usability, and productivity. Cambridge, MA: The MIT Press.

Lavery, D., & Cockton, G. (1997). Cognitive walkthrough usability evaluation materials (Tech. Report 1997-20). Glasgow, UK: University of Glasgow.

Lewis, C. (1997). Cognitive walkthroughs. In M. G. Helander, T. K. Landauer, & P. V. Prabhu (Eds.), Handbook of human-computer interaction (pp. 717-732). Amsterdam: Elsevier Science.

Lewis, C., Polson, P., Wharton, C., & Rieman, J. (1990). Testing a walkthrough methodology for theory-based design of walk-up-and-use interfaces. In CHI '90 Conference Proceedings (pp. 235-242). New York: ACM Press.

Lewis, J. R. (1994). Sample sizes for usability studies: Additional considerations. Human Factors, 36(2), 368-378.

Lund, A. M. (1997). Expert ratings of usability maxims. Ergonomics in Design, 5(3), 15-20.

Lund, A. M. (1998). The need for a standardized set of usability metrics. In Proceedings of the Human Factors and Ergonomics Society 42nd Annual Meeting (pp. 688-691). Santa Monica, CA: Human Factors and Ergonomics Society.

Mack, R., & Montaniz, F. (1994). Observing, predicting, and analyzing usability problems. In J. Nielsen & R. L. Mack (Eds.), Usability inspection methods (pp. 295-339). New York: John Wiley & Sons.

Mack, R. L., & Nielsen, J. (1994). Executive summary. In J. Nielsen & R. L. Mack (Eds.), Usability inspection methods (pp. 1-23). New York: John Wiley & Sons.

Marchetti, R. (1994). Using usability inspections to find usability problems early in the lifecycle. In Pacific Northwest Software Quality Conference (pp. 1-19). Palo Alto, CA: Hewlett-Packard.

May, J., & Barnard, P. (1995). The case for supportive evaluation during design. Interacting with Computers, 7(2), 115-143.

McCreary, F. A. (1996). InTouch usability evaluation (unpublished manuscript). Blacksburg, VA: Virginia Polytechnic Institute and State University.

Meister, D. (1985). Behavioral analysis and measurement methods. New York: John Wiley & Sons.

Meister, D., Andre, T. S., & Aretz, A. J. (1997). System analysis. In T. S. Andre & A. W. Schopper (Eds.), Human factors engineering in system design (pp. 21-55). Wright-Patterson Air Force Base, OH: CSERIAC.

Molich, R., & Nielsen, J. (1990). Improving a human-computer dialogue. Communications of the ACM, 33(3), 338-348.

Nielsen, J. (1990a). Evaluating the thinking aloud technique for use by computer scientists. In H. R. Hartson & D. Hix (Eds.), Advances in human-computer interaction (Vol. 3, pp. 69-82). Norwood, NJ: Ablex.

Nielsen, J. (1990b). Paper versus computer implementations as mockup scenarios for heuristic evaluation. In Proceedings of the IFIP Third International Conference on Human-Computer Interaction - INTERACT '90 (pp. 315-320). Amsterdam: Elsevier Science.

Nielsen, J. (1992). Finding usability problems through heuristic evaluation. In CHI '92 Conference Proceedings (pp. 373-380). New York: ACM Press.

Nielsen, J. (1993). Usability engineering. Boston: Academic Press.

Nielsen, J. (1994a). Enhancing the explanatory power of usability heuristics. In CHI '94 Conference Proceedings (pp. 152-158). New York: ACM Press.

Nielsen, J. (1994b). Heuristic evaluation. In J. Nielsen & R. L. Mack (Eds.), Usability inspection methods (pp. 25-62). New York: John Wiley & Sons.

Nielsen, J., & Landauer, T. K. (1993). A mathematical model of the finding of usability problems. In INTERCHI '93 Conference Proceedings (pp. 206-213). New York: ACM Press.

Nielsen, J., & Molich, R. (1990). Heuristic evaluation of user interfaces. In CHI '90 Conference Proceedings (pp. 249-256). New York: ACM Press.

Norman, D. A. (1986). Cognitive engineering. In D. A. Norman & S. W. Draper (Eds.), User centered system design: New perspectives on human-computer interaction (pp. 31-61). Hillsdale, NJ: Lawrence Erlbaum Associates.

Olson, J., & Olson, G. (1990). The growth of cognitive modeling in human-computer interaction since GOMS. Human Computer Interaction, 5, 221-265.

Olson, J. S., & Moran, T. P. (1996). Mapping the method muddle: Guidance in using methods for user interface design. In M. Rudisill, C. Lewis, P. Polson, & T. MacKay (Eds.), Human-computer interface designs: Success stories, emerging methods, and real-world context (pp. 269-300). San Francisco, CA: Morgan Kaufman.

Polson, P., & Lewis, C. (1990). Theory-based design for easily learned interfaces. Human-Computer Interaction, 5(2), 191-220.

Polson, P., Lewis, C., Rieman, J., & Wharton, C. (1992a). Cognitive walkthroughs: A method for theory-based evaluation of user interfaces. International Journal of Man-Machine Studies, 36, 741-773.

Polson, P., Rieman, J., Wharton, C., & Olson, J. (1992b). Usability inspection methods: Rationale and examples (Tech. Report CU-ICS 92-07). Boulder, CO: University of Colorado.

Rieman, J., Davies, S., Hair, D. C., Esemplare, M., Polson, P. G., & Lewis, C. (1991). An automated cognitive walkthrough. In CHI '91 Conference Proceedings (pp. 427-428). New York: ACM Press.

Rosenbaum, S. (1989). Usability evaluations vs. usability testing: When and why? IEEE Transactions on Professional Communications, 32(4), 210-216.

Rowley, D. E., & Rhoades, D. G. (1992). The cognitive jogthrough: A fast-paced user interface evaluation procedure. In CHI '92 Conference Proceedings (pp. 389-395). New York: ACM Press.

Rubin, J. (1994). Handbook of usability testing. New York: John Wiley & Sons.

Salton, G., & McGill, M. J. (1983). Introduction to modern information retrieval. New York: McGraw-Hill.

Scerbo, M. W. (1995). Usability testing. In J. Weimer (Ed.), Research techniques in human engineering (pp. 72-111). Englewood Cliffs, NJ: Prentice Hall.

Scriven, M. (1967). Methodology of evaluation. In R. Tyler, R. Gagne, & M. Scriven (Eds.), Perspectives of Curriculum Evaluation (pp. 39-83). Chicago: Rand McNally.

Sears, A. (1997). Heuristic walkthroughs: Finding the problems without the noise. International Journal of Human-Computer Interaction, 9(3), 213-234.

Shneiderman, B. (1998). Designing the user interface: Strategies for effective human-computer interaction. (3rd ed.). Reading, MA: Addison-Wesley.

Smith, S. L., & Mosier, J. N. (1986). Guidelines for designing user interface software (MTR-10090). Bedford, MA: Mitre Corp.

Springett, M. (1998). Linking surface error characteristics to root problems in user-based evaluation studies. In Proceedings of the Working Conference on Advanced Visual Interfaces - AVI '98 (pp. 102-113). New York: ACM Press.

Sutcliffe, A., Ryan, M., Springett, M., & Doubleday, A. (1996). Model mismatch analysis: Towards a deeper evaluation of users' usability problems (School of Informatics Report). London: City University.

Swets, J. A. (1964). Signal detection and recognition by human observers. New York: Wiley.

Teubner, A. L., & Vaske, J. J. (1988). Monitoring computer users' behavior in office environments. Behaviour and Information Technology, 7(1), 67-78.

van Rens, L. S. (1997). Usability problem classifier. Unpublished master's thesis, Virginia Polytechnic Institute and State University, Blacksburg, VA.

Verbeek, M., & Van Oostendorp, H. (1998). Adjusting the cognitive walkthrough using the think-aloud method. In Contemporary Ergonomics: Proceedings of the Annual Conference of the Ergonomics Society (pp. 333-337). London, UK: Taylor and Francis.

Virzi, R., Sorce, J., & Herbert, L. B. (1993). A comparison of three usability evaluation methods: Heuristic, think-aloud, and performance testing. In Proceedings of the Human Factors and Ergonomics Society 36th Annual Meeting (pp. 309-313). Santa Monica, CA: Human Factors and Ergonomics Society.

Virzi, R. A. (1990). Streamlining the design process: Running fewer subjects. In Proceedings of the Human Factors and Ergonomics Society 34th Annual Meeting (pp. 291-294). Santa Monica, CA: Human Factors and Ergonomics Society.

Virzi, R. A. (1992). Refining the test phase of usability evaluation: How many subjects is enough? Human Factors, 34(4), 457-468.

Virzi, R. A. (1997). Usability inspection methods. In M. G. Helander, T. K. Landauer, & P. V. Prabhu (Eds.), Handbook of human-computer interaction (2nd ed., pp. 705-715). Amsterdam: Elsevier Science.

Wharton, C. (1992). Cognitive walkthroughs: Instructions, forms, and examples (Tech. Report CU-ICS-92-17). Boulder, CO: University of Colorado.

Wharton, C., Bradford, J., Jeffries, R., & Franzke, M. (1992). Applying cognitive walkthroughs to more complex user interfaces: Experiences, issues, and recommendations. In CHI '92 Conference Proceedings (pp. 381-388). New York: ACM Press.

Wharton, C., Rieman, J., Lewis, C., & Polson, P. (1993). The cognitive walkthrough method: A practitioner's guide (Tech. Report CU-ICS-93-07). Boulder, CO: University of Colorado.

Wharton, C., Rieman, J., Lewis, C., & Polson, P. (1994). The cognitive walkthrough method: A practitioner's guide. In J. Nielsen & R. L. Mack (Eds.), Usability inspection methods (pp. 105-140). New York: John Wiley & Sons.

Whitefield, A., Wilson, F., & Dowell, J. (1991). A framework for human factors evaluation. Behaviour and Information Technology, 10(1), 65-79.

Whiteside, J., Bennett, J., & Holtzblatt, D. (1988). Usability engineering: Our experience and evolution. In M. Helander (Ed.), Handbook of human-computer interaction (pp. 791-817). Amsterdam: Elsevier Science.

Williges, R. C., & Hartson, H. R. (1986). Human-computer dialogue design and research issues. In R. W. Ehrich & R. C. Williges (Eds.), Human-computer dialogue design (pp. 367-376). Amsterdam: Elsevier.

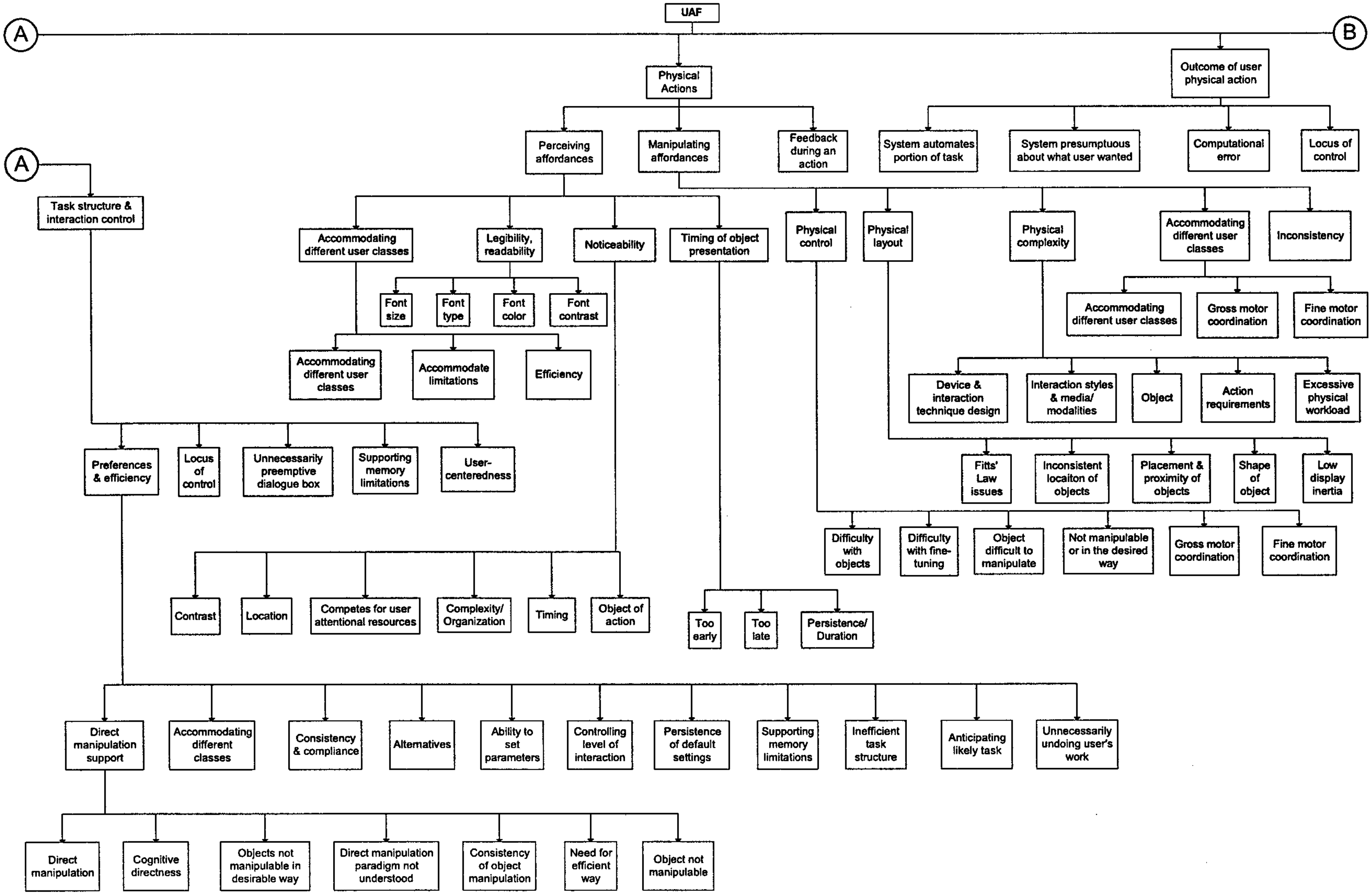
Williges, R. C., Williges, B. H., & Elkerton, J. (1987). Software interface design. In G. Salvendy (Ed.), Handbook of human factors (pp. 1416-1449). New York: Wiley.

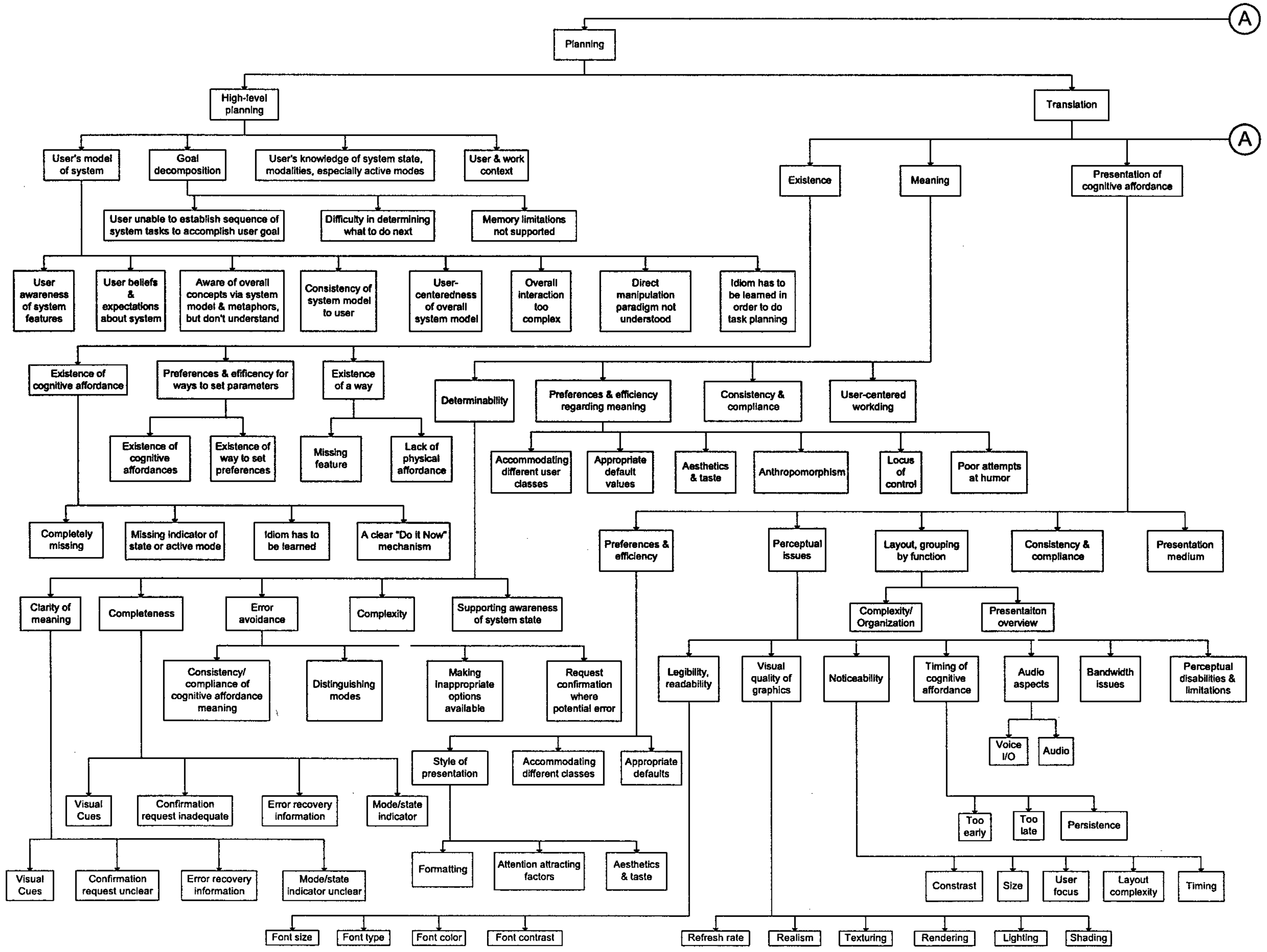
Wixon, D., & Wilson, C. (1997). Chapter 27. The usability engineering framework for product design and evaluation. In M. G. Helander, T. K. Landauer, & P. V. Prabhu (Eds.), Handbook of human-computer interaction (2nd ed., pp. 653-688). Amsterdam: Elsevier Science.

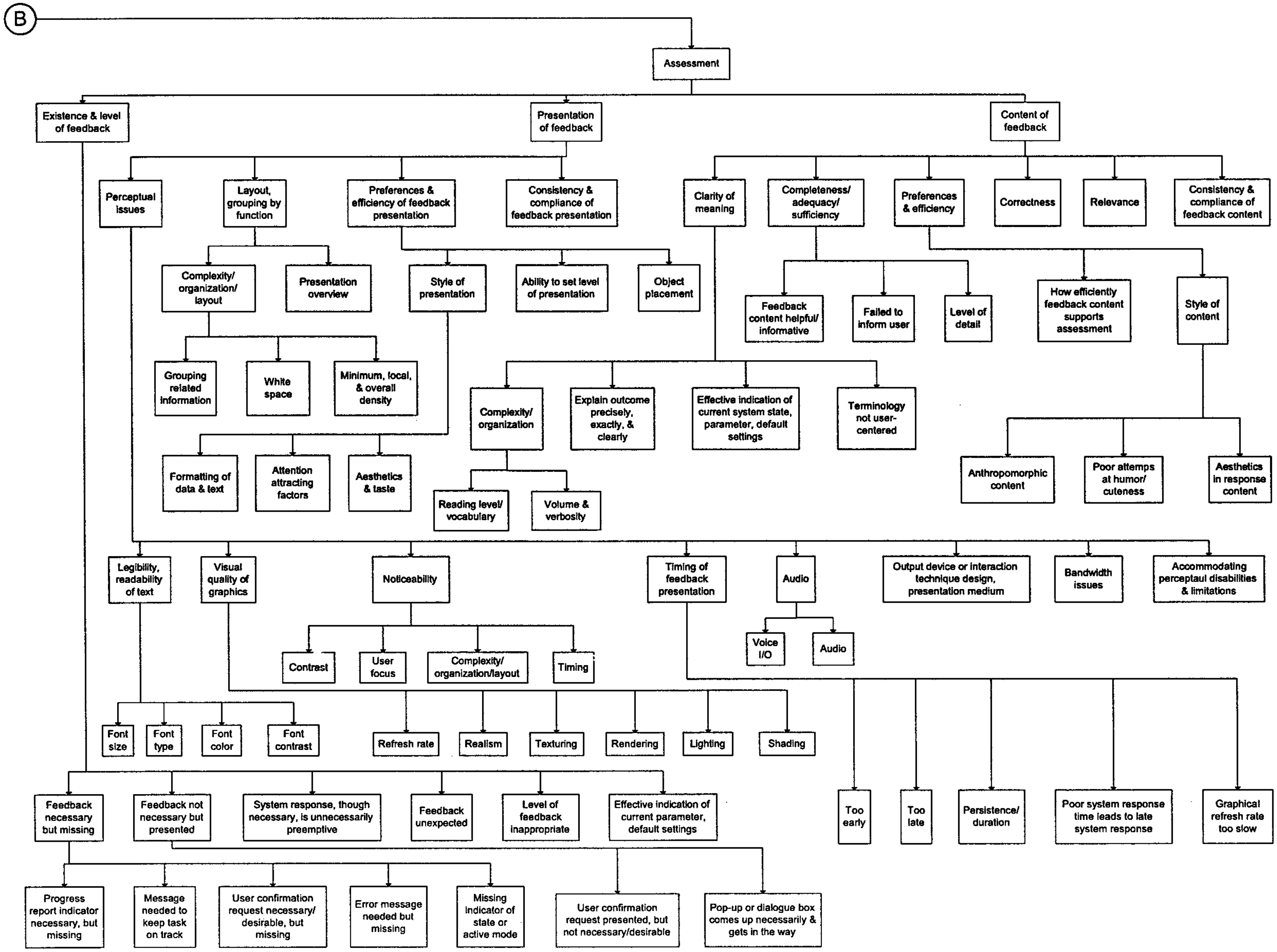
Wright, P., & Monk, A. (1991). A cost-effective evaluation method for designers. International Journal of Man-Machine Studies, 35, 891-912.

Yourdon, E. (1989). Structured walkthroughs. (4th ed.). Englewood Cliffs, NJ: Yourdon Press.

Appendix A.   Detailed Layout of the UAF


UAF
A
B
Physical Actions
Outcome of user physical action
Perceiving affordances
Manipulating affordances
Feedback during an action
System automates portion of task
System presumptuous about what user wanted
Computational error
Locus of control
Task structure & interaction control
Accommodating different user classes
Legibility, readability
Noticeability
Timing of object presentation
Physical control
Physical layout
Physical complexity
Accommodating different user classes
Inconsistency
Font size
Font type
Font color
Font contrast
Accommodating different user classes
Gross motor coordination
Fine motor coordination
Accommodating different user classes
Accommodate limitations
Efficiency
Device & interaction technique design
Interaction styles & media/ modalities
Object
Action requirements
Excessive physical workload
Preferences & efficiency
Locus of control
Unnecessarily preemptive dialogue box
Supporting memory limitations
User-centeredness
Fitts' Law issues
Inconsistent locaiton of objects
Placement & proximity of objects
Shape of object
Low display inertia
Difficulty with objects
Difficulty with fine-tuning
Object difficult to manipulate
Not manipulable or in the desired way
Gross motor coordination
Fine motor coordination
Contrast
Location
Competes for user attentional resources
Complexity/ Organization
Timing
Object of action
Too early
Too late
Persistence/ Duration
Direct manipulation support
Accommodating different classes
Consistency & compliance
Alternatives
Ability to set parameters
Controlling level of interaction
Persistence of default settings
Supporting memory limitations
Inefficient task structure
Anticipating likely task
Unnecessarily undoing user's work
Direct manipulation
Cognitive directness
Objects not manipulable in desirable way
Direct manipulation paradigm not understood
Consistency of object manipulation
Need for efficient way
Object not manipulable


A
Planning
High-level planning
Translation
A
User's model of system
Goal decomposition
User's knowledge of system state, modalities, especially active modes
User & work context
Existence
Meaning
Presentation of cognitive affordance
User unable to establish sequence of system tasks to accomplish user goal
Difficulty in determining what to do next
Memory limitations not supported
User awareness of system features
User beliefs & expectations about system
Aware of overall concepts via system model & metaphors, but don't understand
Consistency of system model to user
User-centeredness of overall system model
Overall interaction too complex
Direct manipulation paradigm not understood
Idiom has to be learned in order to do task planning
Existence of cognitive affordance
Preferences & efficency for ways to set parameters
Existence of a way
Determinability
Preferences & efficiency regarding meaning
Consistency & compliance
User-centered workding
Existence of cognitive affordances
Existence of way to set preferences
Missing feature
Lack of physical affordance
Accommodating different user classes
Appropriate default values
Aesthetics & taste
Anthropomorphism
Locus of control
Poor attempts at humor
Completely missing
Missing indicator of state or active mode
Idiom has to be learned
A clear "Do it Now" mechanism
Preferences & efficiency
Perceptual issues
Layout, grouping by function
Consistency & compliance
Presentation medium
Clarity of meaning
Completeness
Error avoidance
Complexity
Supporting awareness of system state
Complexity/ Organization
Presentaiton overview
Consistency/ compliance of cognitive affordance meaning
Distinguishing modes
Making inappropriate options available
Request confirmation where potential error
Legibility, readability
Visual quality of graphics
Noticeability
Timing of cognitive affordance
Audio aspects
Bandwidth issues
Perceptual disabilities & limitations
Voice I/O
Audio
Style of presentation
Accommodating different classes
Appropriate defaults
Visual Cues
Confirmation request inadequate
Error recovery information
Mode/state indicator
Too early
Too late
Persistence
Visual Cues
Confirmation request unclear
Error recovery information
Mode/state indicator unclear
Formatting
Attention attracting factors
Aesthetics & taste
Constrast
Size
User focus
Layout complexity
Timing
Font size
Font type
Font color
Font contrast
Refresh rate
Realism
Texturing
Rendering
Lighting
Shading


B
Assessment
Existence & level of feedback
Presentation of feedback
Content of feedback
Perceptual issues
Layout, grouping by function
Preferences & efficiency of feedback presentation
Consistency & compliance of feedback presentation
Clarity of meaning
Completeness/ adequacy/ sufficiency
Preferences & efficiency
Correctness
Relevance
Consistency & compliance of feedback content
Complexity/ organization/ layout
Presentation overview
Style of presentation
Ability to set level of presentation
Object placement
Feedback content helpful/ informative
Failed to inform user
Level of detail
How efficiently feedback content supports assessment
Style of content
Grouping related information
White space
Minimum, local, & overall density
Complexity/ organization
Explain outcome precisely, exactly, & clearly
Effective indication of current system state, parameter, default settings
Terminology not user-centered
Formatting of data & text
Attention attracting factors
Aesthetics & taste
Reading level/ vocabulary
Volume & verbosity
Anthropomorphic content
Poor attemps at humor/ cuteness
Aesthetics in response content
Legibility, readability of text
Visual quality of graphics
Noticeability
Timing of feedback presentation
Audio
Output device or interaction technique design, presentation medium
Bandwidth issues
Accommodating perceptaul disabilities & limitations
Contrast
User focus
Complexity/ organization/layout
Timing
Voice I/O
Audio
Font size
Font type
Font color
Font contrast
Refresh rate
Realism
Texturing
Rendering
Lighting
Shading
Feedback necessary but missing
Feedback not necessary but presented
System response, though necessary, is unnecessarily preemptive
Feedback unexpected
Level of feedback inappropriate
Effective indication of current parameter, default settings
Too early
Too late
Persistence/ duration
Poor system response time leads to late system response
Graphical refresh rate too slow
Progress report indicator necessary, but missing
Message needed to keep task on track
User confirmation request necessary/ desirable, but missing
Error message needed but missing
Missing indicator of state or active mode
User confirmation request presented, but not necessary/desirable
Pop-up or dialogue box comes up necessarily & gets in the way

Appendix B. Informed Consent Form for Reliability Study

Virginia Polytechnic Institute and State University
Department of Industrial and Systems Engineering

Informed Consent for Participant of Investigative Project

Title of Project:	User Action Framework Reliability Assessment
Principal Investigators:	Mr. Terence S. Andre Dr. Robert C. Williges, Professor (Co-chair) Dr. H. Rex Hartson, Professor (Co-chair)

I. The Purpose of this Research

You are invited to participate in a study of the User Action Framework (UAF). The UAF is a methodology for organizing usability concepts and issues. This study involves experimentation for the purpose of evaluating and improving the UAF.

II. Procedures

You will be asked to perform a set of tasks using the UAF database. These tasks consist of classifying a set of usability critical incidents using the UAF. Your role in these tests is that of evaluator of the UAF. We are not evaluating you or your performance in any way; you are helping us to evaluate our system. All information that you help us attain will remain anonymous. Your actions will be noted and you will be asked to describe verbally your classification process. You may be asked questions during and after the evaluation, in order to clarify our understanding of your evaluation. The session will last about 2 hours. The tasks are not very tiring, but you are welcome to take rest breaks as needed. If you prefer, the session may be divided into two shorter sessions.

III. Risks

There are minimal risks associated with this study other than those encountered from using a computer and a web-browser in everyday activities.

IV. Benefits of this Project

Your participation in this project will provide information that may be used to improve the UAF. No promise or guarantee of benefits has been made to encourage you to participate. If you would like to receive a synopsis or summary of this research when it is completed, please notify Terence Andre.

V. Extent of Anonymity and Confidentially

The results of this study will be kept strictly confidential. Your written consent is required for the researchers to release any data identified with you as an individual to anyone other than personnel working on the project. The information you provide will have your name removed and only a subject number will identify you during analyses and any written reports of the research.

VI. Compensation

No financial compensation will be offered to you for participation in this project.

VII. Freedom to Withdraw

You are free to withdraw from this study at any time for any reason without penalty.

VIII. Approval of Research

This research has been approved, as required, by the Institutional Review Board for projects involving human subjects at Virginia Polytechnic Institute and State University, and by the Department of Industrial and Systems Engineering.

IX. Participant's Responsibilities

I voluntarily agree to participate in this study. I have the following responsibilities:

- To notify the experimenter at any time about a desire to discontinue participation.
- After completion of this study, I will not discuss my experiences with any other individual for a period of one month. This will ensure that everyone will begin the study with the same level of knowledge and expectations.

X. Participant's Permission

I have read and understand the Informed Consent and conditions of this project. I have had all my questions answered. I hereby acknowledge the above and give my voluntary consent for participation in this project. If I participate, I may withdraw at any time without penalty.

______________________________ ______________________

Signature Date

Should I have any questions about this research or its conduct, I may contact:

Mr. Terence S. Andre Investigator	231-9089
Dr. Robert C. Williges Faculty Advisor	231-6270
Dr. H. Rex Hartson Faculty Advisor	231-4857

In addition, if you have detailed questions regarding your rights as a participant in University research, you may contact the following individual:

Mr. Tom Hurd
Chair, University Institutional Review Board for Research Involving Human Subjects
301 Burruss Hall
Virginia Tech
Blacksburg, VA 24061
(540) 231-5281

Appendix C. Training Materials for the Reliability Study

UAF Reliability Training Materials

TRAINING PACKAGE

The User Action Framework Reliability Study

DIRECTIONS: This training package introduces you to the User Action Framework. The researcher will review this training package with you before you begin the classification process. This training package should take approximately 20-30 minutes to complete. After completing this training package, you will be given the 15 usability problem case descriptions and asked to classify each one using the UAF.

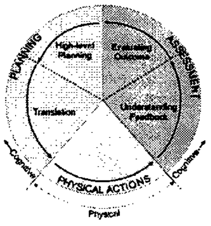
1

What is the UAF?

- ❑ **Conceptual framework of usability concepts and issues**
- ❑ **Formed by combining a user interaction cycle with a knowledge base of usability concepts and issues**
- ❑ **UAF provides a basis for: organizing, discussing, classifying, and reporting usability problems**
- ❑ **Is the basis for a set of usability support methods and tools:**
  - ● Usability Problem Design Guide
  - ● Usability Problem Inspector
  - ● Usability Problem Classifier
  - ● Usability Problem Database

2

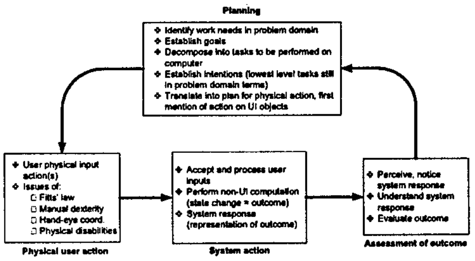
The Interaction Cycle



- ❑ **Helps organize usability issues and concepts**
- ❑ **Adapted from Norman (1986)**
- ❑ **A picture of how interaction happens**
- ❑ **Based on user actions (cognitive and physical)**

3

Flow of Interaction Cycle



Planning

- ✦ Identify work needs in problem domain
- ✦ Establish goals
- ✦ Decompose into tasks to be performed on computer
- ✦ Establish intentions (lowest level tasks still in problem domain terms)
- ✦ Translate into plan for physical action, first mention of action on UI objects

Physical user action

- ✦ User physical input action(s)
- ✦ Issues of:
  - □ Fitts' law
  - □ Manual dexterity
  - □ Hand-eye coord.
  - □ Physical disabilities

System action

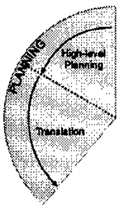
- ✦ Accept and process user inputs
- ✦ Perform non-UI computation (state change = outcome)
- ✦ System response (representation of outcome)

Assessment of outcome

- ✦ Perceive, notice system response
- ✦ Understand system response
- ✦ Evaluate outcome

4

About Planning



- ❑ **Planning breaks down into two important parts:**
  - ● High-level planning
  - ● Translation

Goal: Always work/problem domain (e.g., produce business letter)

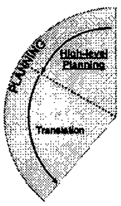
Task: Planning tasks to be done using computer (e.g., formatting the page)

Intention: Planning intentions to be done using computer (e.g., user intends to set left margin)

Action plan: Plan for physical actions to be done on computer (e.g., decide to drag margin marker in MS Word)

5

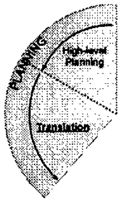
About High-Level Planning



- ❑ Where user decides what to do
- ❑ Identify work needs and establish goals, tasks, and intentions
- ❑ Example areas:
  - ● Goal decomposition (what to do next, understanding sequence of tasks)
  - ● User's model of system (understanding overall system model/metaphor, expectations)

6

About Translation



- ❑ Where user figures out how to do it ("getting started")
- ❑ Translating from the language of the problem domain to the language of actions upon user interface objects
- ❑ Example areas:
  - ● Existence of a way (missing feature)
  - ● Cognitive affordance to show the way (visual cues)
  - ● Efficient way to "do it" (accommodating different user classes, shortcuts)
  - ● Help user do right thing (error avoidance)

7

About Physical Actions

PHYSICAL ACTIONS

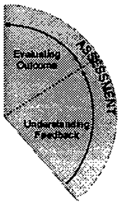
- ❑ All user inputs to operate controls and manipulate objects within the user interface (e.g., clicking, typing, dragging)
- ❑ Example areas:
  - ● Perceiving affordances
  - ● Manipulating affordances
  - ● Physical control
  - ● Fitts' law
  - ● Manual dexterity
  - ● Physical accessibility and disability

8

About Outcome

- ❑ Internal state change within system due to the user action
- ❑ User normally infers the outcome based on system response, through feedback
- ❑ Example areas:
  - ● System automation
  - ● Locus of control
  - ● System is presumptuous about what the user wanted
  - ● System errors

About Assessment



- ❑ Evaluate what happened and the favorability or desirability of the outcome
- ❑ How feedback is perceived, understood, and used to assess the outcome of a user action
- ❑ Example areas:
  - ● Existence of feedback (necessary but missing, unnecessary, not expected)
  - ● Appearance of feedback (legibility, noticeability)
  - ● How well feedback is expressed (clarity, completeness, efficient)

Classifying Problems

- ❑ **Finding the correct entry point in the Interaction Cycle for a usability issue is based on asking:**
  - ● How the user and task performance are affected by the design during interaction
- ❑ **Classification of a usability situation begins by associating it with the appropriate cognitive or physical user action in the Interaction Cycle**
- ❑ **Then the usability situation is classified within the taxonomy underneath the Interaction Cycle by systematically matching usability attributes that pair up effects of a design feature on the user with usability problem causes in the interaction design**

Example

- ❑ **Locate the Cause-in-Design (essence of the problem)**
- ❑ **Example: Hard to read feedback message**

<u>Cause-in-Design</u>

▶ Assessment

  ▶ Presentation

    ▶ Perceptual Issues

      ■ Legibility

Key Terms

- ❑ **Cognitive affordance (visual cues to see a button)**
  - ● Aids for knowing and understanding
  - ● Aids to show the way
- ❑ **Physical affordance (a button that can be "clicked")**
  - ● Aids for doing
- ❑ **Example**
  - ● A chair provides both. Physical affordance of a chair allows sitting on it. Cognitive affordance of a chair lets user see that it is something to sit on
- ❑ **Effective affordances support the users' ability to plan physical actions to carry out intentions**

TRAINING PACKAGE

Heuristics
Reliability Study

September 1999

DIRECTIONS: This packet contains a brief training package to reacquaint you to Heuristic Evaluation for user interfaces. Even if you consider yourself to be an expert user with this technique, please take a moment to review this package before completing the study. This training package should take approximately 20-30 minutes to complete.

IMPORTANT: You will not be able to refer to this training package in conjunction when classifying the test cases.

What is Heuristic Evaluation?

- **Heuristic evaluation is a usability engineering method for finding the usability problems in a user interface design**
  - Helps to structure the critique of a system through the use relatively simple and general principles
- **Heuristics are fairly broad usability principles that help the evaluator find usability problems within a system**
  - These heuristics were developed via examination of several hundred usability problems and distilling the most important general principles (the heuristics)
- **Nielsen's set of 10 usability heuristics will be to classify 15 usability problem case descriptions**

Nielsen's Heuristics

- ❑ **Visibility of system status**

  The system should always keep the user's informed about what is going on, through appropriate feedback within reasonable time

- ❑ **Match between system and the real world**

  The system should speak the users' language, with words, phrases, and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order

- ❑ **User control and freedom**

  Users often choose system functions by mistake and will need a clearly marked 'emergency exit' to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

- ❑ **Consistency and standards**

  Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

- ❑ **Error prevention**

  Even better than good error messages is a careful design which prevents a problem from occurring in the first place.

- ❑ **Recognition rather than recall**

  Make objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

- ❑ **Flexibility and efficiency of use**

  Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

- ❑ **Aesthetic and minimalist design**

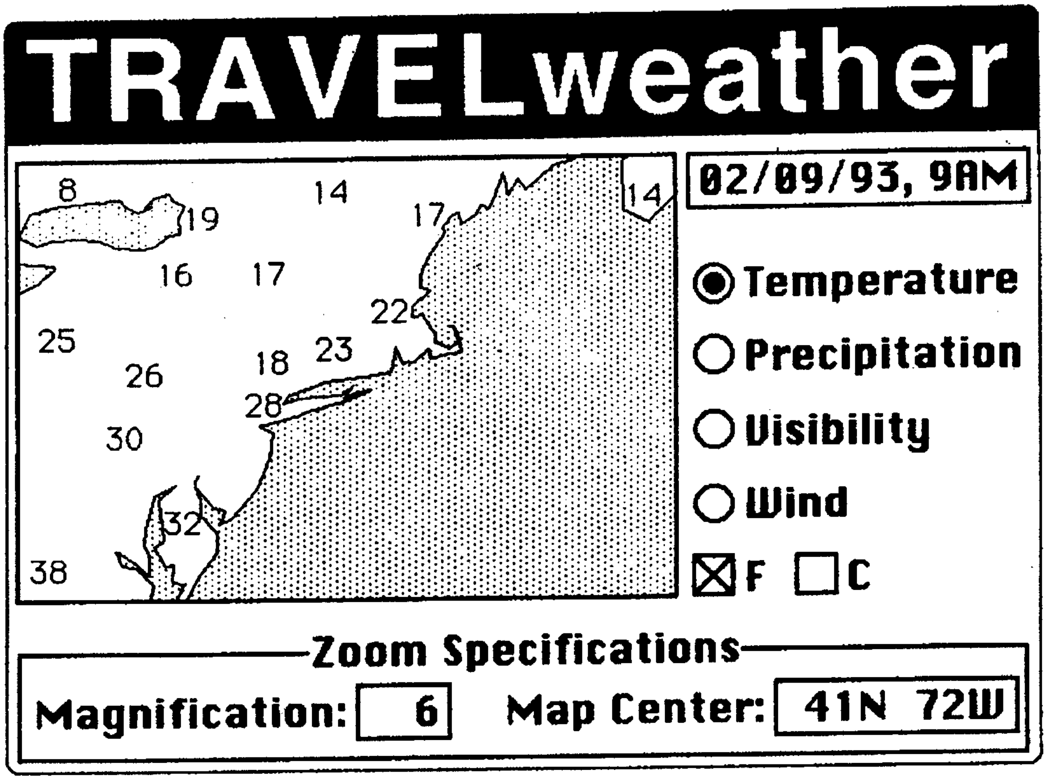
  Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

- ❑ **Help users recognize, diagnose, and recover frcm errors**

  Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

In order to illustrate each of Nielsen's Heuristics, we employ the following heuristic evaluation of a paper mock-up:

Shown below in Figure 1 is the design for a system to provide weather information to travelers. TRAVELweather (a non-existing system) can provide information about the weather at 3AM, 9AM, 3PM, and 9PM for the current day as well as the two next days, using reported readings for past weather and forecasts to predict future weather. The interface is designed for use on a graphical personal computer with a mouse, and will appear in a separate window on the screen.



TRAVELweather

02/09/93, 9AM

● Temperature

○ Precipitation

○ Visibility

○ Wind

☒ F □ C

Zoom Specifications

Magnification: 6 Map Center: 41N 72W

Figure 1. Screen design for a hypothetical system to provide weather information and forecasts to travelers.

The user operates the interface by typing the desired time into the box in the upper right part of the screen. If the user types a date other than today or the next two days, or if the user types a time other than the four times for which information is available, the system will show an alert dialog box with the following error message: "Weather Data Not Available." The only button in the error message box is an "OK" button. Clicking the OK button will make the dialog box go away and will reset the date and time specification to the previous value.

The user changes the map display by editing the boxes for zoom magnification and for the center of the map. The system ensures that only integer numbers can be typed in the map magnification box by simply beeping every time the user presses a non-number key in that box. If the user types anything in the map center box other than a valid set of coordinates (an integer from 0 to 90 followed by the letter N or S followed by an integer from 0 to 179 followed by the letter W or E), the system will show an alert dialog box with the following error message: "Unknown Map Coordinates." The only button in the error message box is an "OK" button. Clicking the OK button will make the dialog box go away and will reset the coordinates to their previous value.

With respect to all three input boxes, the user's changes take effect as soon as the user clicks the mouse outside a box after having edited it.

Based upon this mock-up, we will now show an example of a usability problem that corresponds to each of Nielsen's Heuristics. It is worth noting that not all of the system's shortcomings are discussed. For the purpose of this illustration, only one example was chosen for each heuristic.

Visibility of system status

The system should always keep the user's informed about what is going on, through appropriate feedback within reasonable time

The map should display the names of at least some larger cities and other locations of interest to allow users to better recognize these locations. One way of including additional names without cluttering up the map would be to pop up the names of cities close to the weather stations when the user slides the mouse over a weather reading.

Match between system and the real world

The system should speak the users' language, with words, phrases, and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order

The pattern used to denote oceans and lakes does not make it sufficiently clear what parts of the map are land and what are water. Instead of the current pattern, use a wavy pattern (or blue on a color screen).

User control and freedom

Users often choose system functions by mistake and will need a clearly marked 'emergency exit' to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

Users should not be punished for making errors by having the system delete all their input. Instead, the erroneous user input should be retained to allow the user to edit it. Alternatively, to keep the fields on the main screen correct, repeat the erroneous input in the error dialog box and allow users to edit it there.

Consistency and standards

Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

Error prevention

Even better than good error messages is a careful design which prevents a problem from occurring in the first place.

Requiring the user to click outside the entry box before changes will take effect is error prone. It is likely that many users will forget this and will wonder why nothing happens after they changed the text. One possible way to reduce the likelihood of this error is to have an explicit "do it" button. Also, the user's changes should lake effect if the user hits the enter or return keys. Redesigning the interface as suggested above to replace the text entry boxes with a combination of pop-up menus, scroll bars, zoom buttons, and a click shortcut would also solve the problem.

Recognition rather than recall

Make objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

No options are presented for displaying the appropriate instructions for operating the system.

Flexibility and efficiency of use

Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

It is not clear from the specification to what extent the system will be used repeatedly by the same users (home or office use) or whether it will be used mainly by a flow of changing users (airport etc. use). If the same users can be expected to repeatedly use the system, they will probably also repeatedly ask for weather for the same areas. Support for this user need can be provided by having the system remember the last seven or so locations typed in the map center box and provide direct access to them through a pop-up menu. The next time the system was started, the map could also come up with the zoom specifications (magnification and center) set to the values from the last time the same user used the system.

Aesthetic and minimalist design

Dialogues should not contain information that is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

The name of the system is displayed much too prominently. By making the name smaller, room could be provided for alternative dialogue elements, or the screen could be made less busy.

Help users recognize, diagnose, and recover from errors

Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

The error message "Weather data not available" is not precise. Instead, the system should repeat the date and time as entered by the user and explain why they were not acceptable to the system. Different error messages should be used for dates and times that are not formatted correctly, dates and times that are before or after the time interval for which weather information is available, and times that are not one of the four hours for which information is available.

Help and documentation

Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

No user help or assistance is provided for this system.

TEST CASES

Heuristics
Reliability Study

September 1999

DIRECTIONS: This packet contains fifteen scenarios. For each scenario, a usability problem is identified and described in detail. Unlike the examples presented in the training materials, you will not need to identify multiple problems with a particular scenario. Rather, using Nielsen's Usability Heuristics you are asked to classify the one problem (or aspect of a problem) that we have identified for you.

Specifically, for each usability scenario, A) Briefly describe your understanding of the usability problem conveyed in the above example, and B) Select the primary heuristic that applies. If more than one heuristic applies, rank order them - up to 3 choices - from 1 to 3 with 1 being the most applicable.

An example scenario is provided on the next page.

IMPORTANT: While you are encouraged to refer to the heuristic reference sheet, please ***do not*** refer to the training materials as you complete this exercise.

EXAMPLE
A word processor and email program made by the same company are the main applications used by an individual with extensive clerical responsibilities. Through sheer practice and no formal training, he has become proficient in using both programs; however, he regularly encounters one particular error when using the email program. Sometimes, by sheer habit due to extensive use of the word processing software, he will perform a CTRL-S key press sequence in the middle of a lengthy email message and inadvertently send it before it was completed. The problem is that the word processing software employs the CTRL-S key press sequence to initiate saving the file, while the email software employs the CTRL-S key press to immediately send the current message.

Directions: Based upon the above description, A) Briefly describe your understanding of the usability problem conveyed in the above example, and B) Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

A. Brief description of your understanding of the usability problem described above.

The company has failed to recognize that a lack of consistency/standardization exists between these two products. Also, there is not a 'safety net' or opportunity for graceful recovery designed to protect users from this type of error.

B. Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

Rank Order	Heuristic (see heuristic reference sheet for explanation)
	Visibility of system status
	Match between system and the real world
	User control and freedom
1	Consistency and standards
2	Error prevention
	Recognition rather than recall
	Flexibility and efficiency of use
	Aesthetic and minimalist design
3	Help users recognize, diagnose, and recover from errors
	Help and documentation

A user thinks he knows what he is doing on a certain task, but when he selects an object and clicks on an icon, he gets an error message. The problem is the error message is in a very small font and the color is too close to the background color, so he has difficulty reading the message. This problem is not about getting the error, but about the message received after the error.

Directions: Based upon the above description, A) Briefly describe your understanding of the usability problem conveyed in the above example, and B) Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

A. Brief description of your understanding of the usability problem described above.

B. Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

Rank Order	**Heuristic (see heuristic reference sheet for explanation)**
________	Visibility of system status
________	Match between system and the real world
________	User control and freedom
________	Consistency and standards
________	Error prevention
________	Recognition rather than recall
________	Flexibility and efficiency of use
________	Aesthetic and minimalist design
________	Help users recognize, diagnose, and recover from errors
________	Help and documentation

User knows generally that the Master Document feature of Word is used to allow treating several chapters in different files as a single document (e.g., for global editing). She wants to use this for her multiple thesis chapters, but the system does not help her figure out what she can do with it or how it might help her with her task. She has not yet done anything with it.

Directions: Based upon the above description, A) Briefly describe your understanding of the usability problem conveyed in the above example, and B) Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

A. Brief description of your understanding of the usability problem described above.

B. Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

Rank Order	**Heuristic (see heuristic reference sheet for explanation)**
________	Visibility of system status
________	Match between system and the real world
________	User control and freedom
________	Consistency and standards
________	Error prevention
________	Recognition rather than recall
________	Flexibility and efficiency of use
________	Aesthetic and minimalist design
________	Help users recognize, diagnose, and recover from errors
________	Help and documentation

A user of a personal document retrieval system has been deleting numbered documents. The user now wants to reuse the old document numbers, but the system does not allow this.

Directions: Based upon the above description, A) Briefly describe your understanding of the usability problem conveyed in the above example, and B) Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

A. Brief description of your understanding of the usability problem described above.

B. Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

Rank Order	**Heuristic (see heuristic reference sheet for explanation)**
	Visibility of system status
	Match between system and the real world
	User control and freedom
	Consistency and standards
	Error prevention
	Recognition rather than recall
	Flexibility and efficiency of use
	Aesthetic and minimalist design
	Help users recognize, diagnose, and recover from errors
	Help and documentation

A drawing package has a very large number of functions, most of which are available to the user via a button bar crammed with various buttons. Each function is accessible via another way (e.g., a menu choice) as well, and our observations tell us that users mainly use the button bar for familiar and frequently used functions. So they don't usually have trouble figuring out which button to use; if it's not a familiar function, they don't use the buttons. This problem is about what happens when they do use the buttons. Because there are so many buttons, they are somewhat small and crowded together. This, combined with the fast action of experienced users, leads to clicking on the wrong button more often than users would like.

Directions: Based upon the above description, A) Briefly describe your understanding of the usability problem conveyed in the above example, and B) Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

A. Brief description of your understanding of the usability problem described above.

B. Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

Rank Order	**Heuristic (see heuristic reference sheet for explanation)**
________	Visibility of system status
________	Match between system and the real world
________	User control and freedom
________	Consistency and standards
________	Error prevention
________	Recognition rather than recall
________	Flexibility and efficiency of use
________	Aesthetic and minimalist design
________	Help users recognize, diagnose, and recover from errors
________	Help and documentation

A person using a Windows program for ftp file transfer wants to rename a file. She selects the file name and tries to type over it (as she does in the Explorer program on her PC), but this does not work. Eventually she figures out that you have to select the filename and then click the Rename command from a button bar. That leads to a small dialogue box with the filename in a text field where she can edit the name and click on OK. When she clicks OK, the system puts the new filename back into the list. She completes the task wondering why the system didn't provide a way to do the task directly.

Directions: Based upon the above description, A) Briefly describe your understanding of the usability problem conveyed in the above example, and B) Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

A. Brief description of your understanding of the usability problem described above.

B. Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

Rank Order	**Heuristic (see heuristic reference sheet for explanation)**
________	Visibility of system status
________	Match between system and the real world
________	User control and freedom
________	Consistency and standards
________	Error prevention
________	Recognition rather than recall
________	Flexibility and efficiency of use
________	Aesthetic and minimalist design
________	Help users recognize, diagnose, and recover from errors
________	Help and documentation

A user of a dbase-family database application had been deleting lots of records in a large database. The user knew that, in dbase applications, "deleted" records are really only marked for deletion (and can be undeleted) until a Pack operation is performed, permanently removing all records marked for deletion. At some point, the user did the Pack operation, but it didn't seem to work. After waiting what seemed like a long time (about 10 seconds), the user pushed the "Escape" key to get back control of the computer. As it turns out, the system *was* doing a Pack operation, which takes a long time for a large database. The user may have interrupted the operation with the "Escape" key, leaving things in an indeterminate state. If the system had let the user know it was, in fact, doing the requested Pack operation, he would have waited for it to complete.

Directions: Based upon the above description, A) Briefly describe your understanding of the usability problem conveyed in the above example, and B) Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

A. Brief description of your understanding of the usability problem described above.

B. Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

Rank Order	**Heuristic (see heuristic reference sheet for explanation)**
________	Visibility of system status
________	Match between system and the real world
________	User control and freedom
________	Consistency and standards
________	Error prevention
________	Recognition rather than recall
________	Flexibility and efficiency of use
________	Aesthetic and minimalist design
________	Help users recognize, diagnose, and recover from errors
________	Help and documentation

A database user accidentally deleted a number of related records. She knows that she can back out of this and correct the error, but the system does not help her find a way to do it. There is a button, labeled 'Back' for recovery from deletion but she was looking for something like 'Undelete' and didn't make the connection.

Directions: Based upon the above description, A) Briefly describe your understanding of the usability problem conveyed in the above example, and B) Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

A. Brief description of your understanding of the usability problem described above.

B. Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

Rank Order	**Heuristic (see heuristic reference sheet for explanation)**
________	Visibility of system status
________	Match between system and the real world
________	User control and freedom
________	Consistency and standards
________	Error prevention
________	Recognition rather than recall
________	Flexibility and efficiency of use
________	Aesthetic and minimalist design
________	Help users recognize, diagnose, and recover from errors
________	Help and documentation

A user of a home design program was ready to print his design and typed a control-P key combination, but it didn't work. After a pause to reflect, the user pulled down the File menu and saw the Print choice and, indeed, there was no "Ctrl+P" next to it. This problem is not about knowing what happened (or didn't), but it's about the system not allowing the user to perform the task the way she wanted.

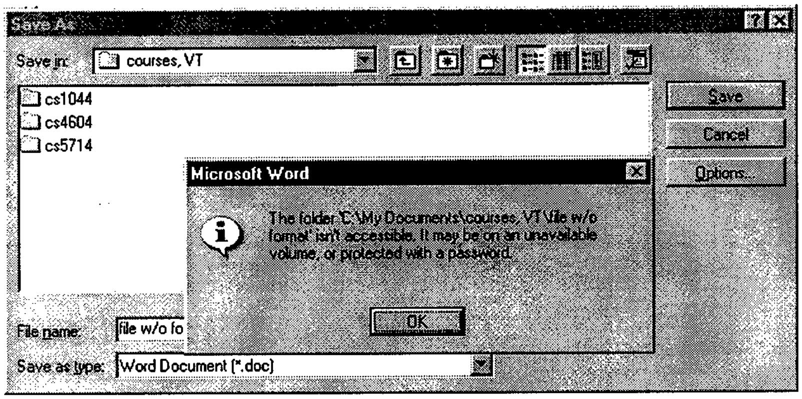
Directions: Based upon the above description, A) Briefly describe your understanding of the usability problem conveyed in the above example, and B) Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

A. Brief description of your understanding of the usability problem described above.

B. Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

Rank Order	**Heuristic (see heuristic reference sheet for explanation)**
________	Visibility of system status
________	Match between system and the real world
________	User control and freedom
________	Consistency and standards
________	Error prevention
________	Recognition rather than recall
________	Flexibility and efficiency of use
________	Aesthetic and minimalist design
________	Help users recognize, diagnose, and recover from errors
________	Help and documentation

A user of Word has created a document that contains an outline of something. As the user saves it (with Save As), she names it "file w/o format" to distinguish it as the unformatted version. She was surprised by the resulting message: "Microsoft Word (with an "I" for "Information", not an "X" indicating an error) – The folder 'C:\My Documents\misc\file w/o format.doc' is not accessible. It may be on an unavailable volume, or protected with a password." This message just seems so far off base that it's hard to make any sense out of it. It seems something is wrong and the file apparently has not been saved yet, but the system does not provide a way to tell what is wrong. (Aside: The real problem is that the file name used contains a "/" character, not allowed in Windows. This apparently made it look like part of a path name to Windows and it went off on the wrong track, confusing the user.)



Save in: courses, VT

The folder 'C:\My Documents\courses, VT\file w/o format' isn't accessible. It may be on an unavailable volume, or protected with a password.

File name: file w/o fo

Save as type: Word Document (*.doc)

Directions: Based upon the above description, A) Briefly describe your understanding of the usability problem conveyed in the above example, and B) Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

A. Brief description of your understanding of the usability prublem described above.

B. Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

Rank Order	**Heuristic (see heuristic reference sheet for explanation)**
________	Visibility of system status
________	Match between system and the real world
________	User control and freedom
________	Consistency and standards
________	Error prevention
________	Recognition rather than recall
________	Flexibility and efficiency of use
________	Aesthetic and minimalist design
________	Help users recognize, diagnose, and recover from errors
________	Help and documentation

In an apparent attempt to be complete, an error message is long; it just carries on. The extra words and explanation obscures the simple message that the system is trying to convey to the user. The user ends up being confused and irritated.

Directions: Based upon the above description, A) Briefly describe your understanding of the usability problem conveyed in the above example, and B) Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

A. Brief description of your understanding of the usability problem described above.

B. Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

Rank Order	**Heuristic (see heuristic reference sheet for explanation)**
________	Visibility of system status
________	Match between system and the real world
________	User control and freedom
________	Consistency and standards
________	Error prevention
________	Recognition rather than recall
________	Flexibility and efficiency of use
________	Aesthetic and minimalist design
________	Help users recognize, diagnose, and recover from errors
________	Help and documentation

User clicks on a button to get the system to carry out a function and a confirmation message appears, "Are you sure you want to xyz?" As this was the only logical operation to the user at this point in the task, the user complained that the confirmation message was unnecessary and irritating and the system had forced the user to make an extra mouse click to deal with it.

Directions: Based upon the above description, A) Briefly describe your understanding of the usability problem conveyed in the above example, and B) Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

A. Brief description of your understanding of the usability problem described above.

B. Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

Rank Order	**Heuristic (see heuristic reference sheet for explanation)**
________	Visibility of system status
________	Match between system and the real world
________	User control and freedom
________	Consistency and standards
________	Error prevention
________	Recognition rather than recall
________	Flexibility and efficiency of use
________	Aesthetic and minimalist design
________	Help users recognize, diagnose, and recover from errors
________	Help and documentation

In a word processing task, while trying to fit a small document just on one page, the user selected all the text and went to the font size menu. Seeing only 10 point and 12 point choices on the menu, she picked 10 but that made the document a bit too small on the page. She commented that it would have been nice if an 11 point font had been available and went on to other parts of the task. The evaluator/observer noted that, in fact, an 11 point font could have been selected, by typing '11' into the little text box part of the menu where the current font size selection is shown, but the system did nothing to help the user know about this possibility.

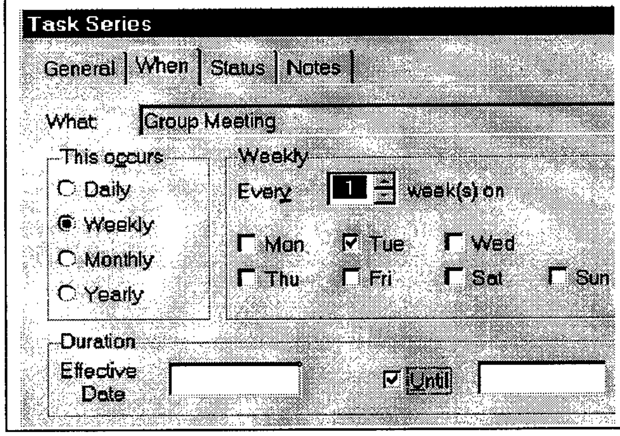
Directions: Based upon the above description, A) Briefly describe your understanding of the usability problem conveyed in the above example, and B) Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

A. Brief description of your understanding of the usability problem described above.

B. Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

Rank Order	**Heuristic (see heuristic reference sheet for explanation)**
________	Visibility of system status
________	Match between system and the real world
________	User control and freedom
________	Consistency and standards
________	Error prevention
________	Recognition rather than recall
________	Flexibility and efficiency of use
________	Aesthetic and minimalist design
________	Help users recognize, diagnose, and recover from errors
________	Help and documentation

User is filling out an on-line form and gets to a field for a date, but there is no indication about what format to use. The user tries something and gets an error message and then is able to correct it and get the system to recognize the date. Some affordance is given through the label that says it is a date value, but it still does not allow her to get the format right the first time.



Task Series

General | When | Status | Notes

What: Group Meeting

This occurs

○ Daily
● Weekly
○ Monthly
○ Yearly

Weekly

Every 1 week(s) on

□ Mon ☑ Tue □ Wed
□ Thu □ Fri □ Sat □ Sun

Duration

Effective Date

☑ Until

Directions: Based upon the above description, A) Briefly describe your understanding of the usability problem conveyed in the above example, and B) Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

A. Brief description of your understanding of the usability problem described above.

B. Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

Rank Order	Heuristic (see heuristic reference sheet for explanation)
________	Visibility of system status
________	Match between system and the real world
________	User control and freedom
________	Consistency and standards
________	Error prevention
________	Recognition rather than recall
________	Flexibility and efficiency of use
________	Aesthetic and minimalist design
________	Help users recognize, diagnose, and recover from errors
________	Help and documentation

In Word on the PC, when you use drag and drop to move text and have to go outside the text that shows on the screen, it scrolls when you get to the top or bottom. Unfortunately, the speed of scrolling is controlled only by the speed of the machine and ends up being too fast for the user to control. The result is thoroughly intimidating and frustrating. The system has put him in a difficult spot, having to hold the mouse button depressed, with the text attached to the cursor, going back and forth not able to find a place to put it.

Directions: Based upon the above description, A) Briefly describe your understanding of the usability problem conveyed in the above example, and B) Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

A. Brief description of your understanding of the usability problem described above.

B. Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

Rank Order	**Heuristic (see heuristic reference sheet for explanation)**
________	Visibility of system status
________	Match between system and the real world
________	User control and freedom
________	Consistency and standards
________	Error prevention
________	Recognition rather than recall
________	Flexibility and efficiency of use
________	Aesthetic and minimalist design
________	Help users recognize, diagnose, and recover from errors
________	Help and documentation

A vision-impaired user has difficulty reading the message text in a dialogue box. This is a more or less expected occurrence for a visually impaired user and is not the problem in itself. The problem here is that the system does not help the user find a way to set the font to be larger and bolder or to get an alternative audio version of the messages.

Directions: Based upon the above description, A) Briefly describe your understanding of the usability problem conveyed in the above example, and B) Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

A. Brief description of your understanding of the usability problem described above.

B. Rank order the top three heuristics from 1 to 3 with 1 being the most applicable.

Rank Order	**Heuristic (see heuristic reference sheet for explanation)**
________	Visibility of system status
________	Match between system and the real world
________	User control and freedom
________	Consistency and standards
________	Error prevention
________	Recognition rather than recall
________	Flexibility and efficiency of use
________	Aesthetic and minimalist design
________	Help users recognize, diagnose, and recover from errors
________	Help and documentation

Appendix D. Informed Consent Form for Lab-Based Usability Test

Virginia Polytechnic Institute and State University
Department of Industrial and Systems Engineering

Informed Consent for Participant of Investigative Project

Title of Project:	Lab-Based Usability Test of InTouch Interface
Principal Investigators:	Mr. Terence S. Andre Dr. Robert C. Williges, Professor (Co-chair) Dr. H. Rex Hartson, Professor (Co-chair)

I. The Purpose of this Research

You are invited to participate in a usability evaluation of a commercial Address Book software program. This study involves usability evaluation for the purpose of generating a list of potential problems with the user interface.

II. Procedures

You will be asked to perform a set of tasks using the Address Book program. Your role in these tests is that of a first-time user of the Address Book program. We are not evaluating you or your performance in any way; you are helping us to evaluate the Address Book program. All information that you help us attain will remain anonymous. Your actions will be noted and you will be asked to clarify interface problems you encounter. You may be asked questions during and after the evaluation, in order to clarify our understanding of your interaction with the Address Book program. The session will last about 1 ½ hours. The tasks are not very tiring, but you are welcome to take rest breaks as needed.

III. Risks

There are minimal risks associated with this study other than those encountered from using a computer and an Address Book program in everyday activities.

IV. Benefits of this Project

Your participation in this project will provide information that will be used in future studies of the Address Book program. There is no direct benefit to you associated with participation in this study other than the extra credit awarded for voluntary participation. You will have the opportunity to see first-hand how a lab-based usability evaluation is conducted. If you would like to receive a synopsis or summary of this research when it is completed, please notify Terence Andre.

V. Extent of Anonymity and Confidentially

The results of this study will be kept strictly confidential. Your written consent is required for the researchers to release any data identified with you as an individual to anyone other than personnel working on the project. The information you provide will have your name removed and only a subject number will identify you during analyses and any written reports of the research.

VI. Compensation

There is no compensation for your participation in this study other than the extra credit identified in the CS 3724 course.

VII. Freedom to Withdraw

You are free to withdraw from this study at any time for any reason without penalty.

VIII. Approval of Research

This research has been approved, as required, by the Institutional Review Board for projects involving human subjects at Virginia Polytechnic Institute and State University, and by the Computer Science Department.

IX. Participant's Responsibilities

I voluntarily agree to participate in this study. I have the following responsibilities:

- To notify the experimenter at any time about a desire to discontinue participation.
- After completion of this study, I will not discuss my experiences with any other individual for a period of one month. This will ensure that everyone will begin the study with the same level of knowledge and expectations.

X. Participant's Permission

I have read and understand the Informed Consent and conditions of this project. I have had all my questions answered. I hereby acknowledge the above and give my voluntary consent for participation in this project. If I participate, I may withdraw at any time without penalty.

______________________________ ______________________

Signature Date

Should I have any questions about this research or its conduct, I may contact:

Mr. Terence S. Andre Investigator	231-9089
Dr. Robert C. Williges Faculty Advisor	231-6270
Dr. H. Rex Hartson Faculty Advisor	231-4857

In addition, if you have detailed questions regarding your rights as a participant in University research, you may contact the following individual:

Mr. Tom Hurd
Chair, University Institutional Review Board for Research Involving Human Subjects
301 Burruss Hall
Virginia Tech
Blacksburg, VA 24061
(540) 231-5281

Appendix E. Pre-Test Questionnaire for Lab-Based Usability Test

InTouch Pre-Test Questionnaire

Directions:

Please rate each of the items according to the scales provided and fill in the appropriate bubble.

1. Age (years): ______________ Sex: ○ Male ○ Female

2. Academic Level: ○ Freshman ○ Sophomore ○ Junior ○ Senior ○ Masters ○ PhD

3. Major of Study: ____________________________________

4. For how long have you been using computers *(please check one)*:
   ○ less than 6 months
   ○ between 6 months and a year
   ○ 1 - 3 years
   ○ 3 years or more

5. Rate your experience with PCs

○	○	○	○	○	○	○	○	○	○
1	2	3	4	5	6	7	8	9	10
No Experience									Very Experienced

6. Rate your experience with MACs

○	○	○	○	○	○	○	○	○	○
1	2	3	4	5	6	7	8	9	10
No Experience									Very Experienced

7. Have you used InTouch© before? ○ Yes ○ No

8. If yes, please indicate how long have you used InTouch©? _______ months

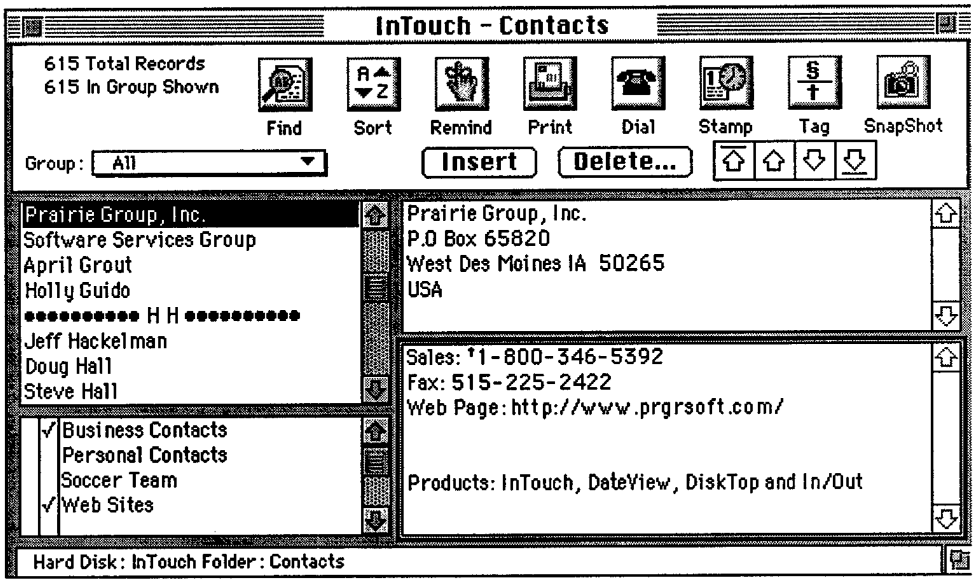
9. Please indicate any other Address Book programs that you have used *(please check one)*:
   ○ Address Book (Broderbund)
   ○ My AddressBook (My Software)
   ○ Outlook Express
   ○ Outlook
   ○ Now Up-to-Date
   ○ Day-Timer Address Book
   ○ Use only a hardcopy Address Book
   ○ Other *(specify)*: ______________________

10. How often do you use an electronic address book program?

○ Every day ○ A few times a week ○ A few times a month ○ Never

Appendix F. Participant Instructions for Lab-Based Usability Test

InTouch Information Sheet for Lab-Based Usability Test Participants



615 Total Records
615 In Group Shown

Prairie Group, Inc.
Software Services Group
April Grout
Holly Guido
●●●●●●●●●● H H ●●●●●●●●●●
Jeff Hackelman
Doug Hall
Steve Hall

✓ Business Contacts
Personal Contacts
Soccer Team
✓ Web Sites

Prairie Group, Inc.
P.O Box 65820
West Des Moines IA 50265
USA

Sales: *1-800-346-5392
Fax: 515-225-2422
Web Page: http://www.prgrsoft.com/

Products: InTouch, DateView, DiskTop and In/Out

Hard Disk: InTouch Folder: Contacts

InTouch is an Address Book...

InTouch is a free-form information manager. Instead of separate fields for each piece of information, InTouch uses just two data entry fields. Most people use InTouch as a contact manager and use the first field for names and addresses and the second field for phone numbers and notes. The second field can hold up to 14 pages of notes that can be time and date stamped to allow you to track conversations, projects or other notations.

InTouch is Free-form...

InTouch offers a refreshing alternative to predefined structured personal information managers or "design your own" databases for keeping information organized with your Macintosh. Sophisticated, but simple to use, InTouch offers a very straight forward way to put information to use.

Free-form is better...

The big advantage to free-form is that you can enter data very rapidly, and InTouch gladly takes what ever information you type. You don't have to worry about how long an address is or how many lines it takes. Foreign addresses present no problem for InTouch. You simply use as many lines as you need, in the order that makes sense to you.

Instructions for Lab-Based Usability Test Participants

1. You will be performing a series of tasks. Tasks should be performed in the order listed.

2. Please read aloud each task description before attempting the task.

3. If the task is not clear or you are not sure how to perform the task using InTouch, please ask the experimenter.

Task 1. Insert a Record

The file that you see is your family rolodex. Put your name, address, and home phone number in the rolodex so other family members can contact you in an emergency.

Task 2. Save a File under a different name

Save a copy of your family rolodex for personal use. Call the copy "MY ROLODEX". You will work with "MY ROLODEX" for the remainder of this session.

Task 3. Find a specific Record

Ken Landry is already entered in your rolodex. He just called you with his new home phone number, add it to his entry. Ken's home phone number is 500-1234.

Task 4. Sort a File

Order the entries in your rolodex alphabetically so that people with the same last name appear together, with family members listed alphabetically by first name.

Task 5. Make a new Group

You notice that all your relatives have "RELATIVE" in the notes field. You decide to make "family" a new group, and make all your relatives members of the group.

Task 6. Import data from a File

You're in charge of organizing soccer this year. Last year's organizer gave you a tab-delimited text file ("Soccer People") containing the names and addresses of the relevant people. Add these people to your rolodex -- put them in the Soccer group for easy reference.

Appendix G. InTouch Post-Test Questionnaire

InTouch Post-Test Questionnaire

Directions:

Please indicate how strongly you disagree or agree to the statements using the scale provided.

	strongly disagree	disagree	neutral	agree	strongly agree	NA
1. It was simple to use InTouch.	○	○	○	○	○	○
2. I am able to complete my tasks quickly using InTouch.	○	○	○	○	○	○
3. I am able to efficiently complete my tasks using InTouch.	○	○	○	○	○	○
4. InTouch gives appropriate feedback to clearly tell me how to fix problems.	○	○	○	○	○	○
5. Whenever I make a mistake using InTouch, I recover easily and quickly.	○	○	○	○	○	○
6. It is easy to use the menus in the InTouch program.	○	○	○	○	○	○
7. It is easy to find the information I need in the InTouch program.	○	○	○	○	○	○
8. I like using InTouch.	○	○	○	○	○	○
9. InTouch has all the functions and capabilities that I expect it to have.	○	○	○	○	○	○

List the most **negative** aspect(s) of the InTouch program:

1. ______________________________

2. ______________________________

3. ______________________________

List the most **positive** aspect(s) of the InTouch program:

1. ______________________________

2. ______________________________

3. ______________________________

Appendix H. Informed Consent Form for Comparison Study

Virginia Polytechnic Institute and State University
Department of Industrial and Systems Engineering

Informed Consent for Participant of Investigative Project

Title of Project:	Comparison Study of Usability Inspection Methods
Principal Investigators:	Mr. Terence S. Andre Dr. Robert C. Williges, Professor (Co-chair) Dr. H. Rex Hartson, Professor (Co-chair)

I. The Purpose of this Research

Usability evaluation is an important part of iterative design and allows the designer to refine design concepts before a formal product is released. The purpose of this project is to conduct a usability analysis of an Address Book program that allows users to keep track of business and personal contact information. The usability analysis will be conducted to determine potential problems that may surface for a typical user interacting with the Address Book program. Usability problems will be classified according to type and severity for design decisions.

II. Procedures

Your task is to play the role of a user and "walkthrough" the interface trying to execute a given set of representative tasks. Along the way, you will need to document your discovery of usability problems on paper and computer forms. The total time for the experiment will take approximately two hours. The first hour will be dedicated to training in the detection and documentation of usability problems. Your analysis will be combined with other evaluators to generate a list of usability problems associated with the Address Book program.

III. Risks

There are minimal risks associated with this study other than those encountered from using a computer and an Address Book program in everyday activities.

IV. Benefits of this Project

Your participation in this project will provide information that will be used in future studies of the Address Book program. There are no direct benefits to you from this research. No promise or guarantee of benefits have been made to encourage you to participate. If you would like to receive a synopsis or summary of this research when it is completed, please notify Terence Andre.

V. Extent of Anonymity and Confidentially

The results of this study will be kept strictly confidential. Your written consent is required for the researchers to release any data identified with you as an individual to anyone other than personnel working on the project. The information you provide will have your name removed and only a subject number will identify you during analyses and any written reports of the research.

VI. Compensation

No financial compensation will be offered to you for participation in this project.

VII. Freedom to Withdraw

You are free to withdraw from this study at any time for any reason without penalty.

VIII. Approval of Research

This research has been approved, as required, by the Institutional Review Board for projects involving human subjects at Virginia Polytechnic Institute and State University, and by the Department of Industrial and Systems Engineering.

IX. Participant's Responsibilities

I voluntarily agree to participate in this study. I have the following responsibilities:

- To notify the experimenter at any time about a desire to discontinue participation.
- After completion of this study, I will not discuss my experiences with any other individual for a period of one month. This will ensure that everyone will begin the study with the same level of knowledge and expectations.

X. Participant's Permission

I have read and understand the Informed Consent and conditions of this project. I have had all my questions answered. I hereby acknowledge the above and give my voluntary consent for participation in this project. If I participate, I may withdraw at any time without penalty.

______________________________ ____________________
Signature Date

Should I have any questions about this research or its conduct, I may contact:

Mr. Terence S. Andre Investigator	231-9089
Dr. Robert C. Williges Faculty Advisor	231-6270
Dr. H. Rex Hartson Faculty Advisor	231-4857

In addition, if you have detailed questions regarding your rights as a participant in University research, you may contact the following individual:

Mr. Tom Hurd
Chair, University Institutional Review Board for Research Involving Human Subjects
301 Burruss Hall
Virginia Tech
Blacksburg, VA 24061
(540) 231-5281

Appendix I. Comparison Study Pre-Test Questionnaire

Comparison Study Pre-Test Questionnaire

1. Age (years): ______________ Sex: ○ Male ○ Female

2. Academic Level: ○ Bachelors ○ Masters ○ PhD Major:_________________________

3. Current Job Title: ____

4. How long have you been in your current position?

   __________ years __________ months

5. Specialty?

   ○ Testing/Evaluation ○ Design ○ Management ○ Research

   ○ Teaching/Academic ○ Other___________________________________

6. How much experience do you have in the specialty selected above?

   __________ years __________ months

7. How often do you perform the following activities:

	Never	About once per year	Few times per year	Few times per month	Few times per week	Daily
Usability testing/observation	○	○	○	○	○	○
Interface design	○	○	○	○	○	○
Using the cognitive walkthrough	○	○	○	○	○	○
Using heuristic evaluation technique	○	○	○	○	○	○

8. Have you used InTouch© before? ○ Yes ○ No

   If yes, please indicate how long have you used InTouch©? ______ months

9. Please indicate any other Address Book programs that you have used *(please check one)*:
   - ○ Address Book (Broderbund)
   - ○ My AddressBook (My Software)
   - ○ Outlook Express
   - ○ Outlook
   - ○ Now Up-to-Date
   - ○ Day-Timer Address Book
   - ○ Use only a hardcopy Address Book
   - ○ Other *(specify)*: _____________________

10. How often do you use an electronic address book program?

Never	About once per year	Few times per year	Few times per month	Few times per week	Daily
○	○	○	○	○	○

Appendix J. Training Materials for UPI Method

Training Package for UPI Method

TRAINING PACKAGE

The Usability Problem Inspector (UPI)

DIRECTIONS: This training package introduces you to the Usability Problem Inspector. The researcher will review this training package with you before you begin the inspection process. This training package should take approximately 20-30 minutes to complete. After completing this training package, you will be introduced to the interface application and instructed on how to begin your inspection of the interface using the Usability Problem Inspector.

1

What is the UPI?

- ❑ **An inspection tool to help evaluators discover and report potential problems with an interface design**
- ❑ **The UPI is primarily task-based, helping the evaluator to note relevant problems**
- ❑ **The UPI is built from an organizing framework – The User Action Framework (UAF)**

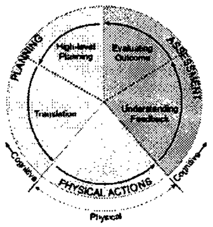
2

The User Action Framework (UAF)

- ❑ **Conceptual framework of usability concepts and issues**
- ❑ **Formed by combining a user interaction cycle with a knowledge base of usability concepts and issues**
- ❑ **UAF provides a basis for: organizing, discussing, classifying, and reporting usability problems**
- ❑ **Is the basis for a set of usability support methods and tools:**
  - ● Usability Problem Design Guide
  - ● Usability Problem Inspector
  - ● Usability Problem Classifier
  - ● Usability Problem Database

3

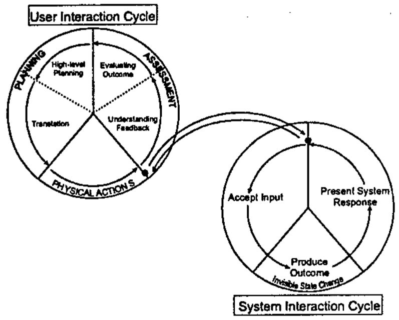
The User Interaction Cycle



- ❑ **Helps organize usability issues and concepts**
- ❑ **Adapted from Norman (1986)**
- ❑ **A picture of how interaction happens**
- ❑ **Based on user actions (cognitive and physical)**

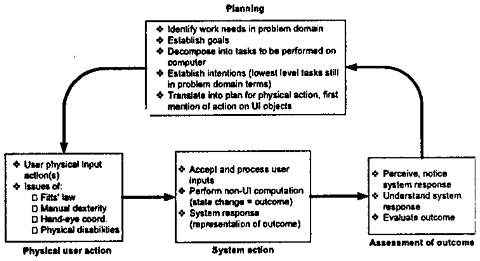
4

User & System Cycle



5

Flow of the Interaction Cycle



Planning

- ✧ Identify work needs in problem domain
- ✧ Establish goals
- ✧ Decompose into tasks to be performed on computer
- ✧ Establish intentions (lowest level tasks still in problem domain terms)
- ✧ Translate into plan for physical action, first mention of action on UI objects

Physical user action

- ✧ User physical input action(s)
- ✧ Issues of:
  - □ Fitts' law
  - □ Manual dexterity
  - □ Hand-eye coord.
  - □ Physical disabilities

System action

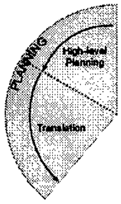
- ✧ Accept and process user inputs
- ✧ Perform non-UI computation (state change = outcome)
- ✧ System response (representation of outcome)

Assessment of outcome

- ✧ Perceive, notice system response
- ✧ Understand system response
- ✧ Evaluate outcome

6

About Planning

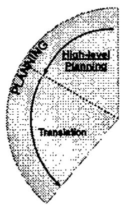


- ❑ **Planning breaks down into two important parts:**
  - ● High-level planning
  - ● Translation

Goal:	Always work/problem domain (e.g., produce business letter)
Task:	Planning tasks to be done using computer (e.g., formatting the page)
Intention:	Planning intentions to be done using computer (e.g., user intends to set left margin)
Action plan:	Plan for physical actions to be done on computer (e.g., decide to drag margin marker in MS Word)

7

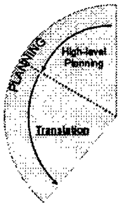
About High-Level Planning



- ❑ Where user decides what to do
- ❑ Identify work needs and establish goals, tasks, and intentions
- ❑ Example areas:
  - ● User's model of system (understanding overall system model/metaphor, expectations)
  - ● Goal decomposition (what to do next, understanding sequence of tasks)

8

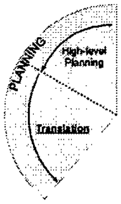
About Translation



- Where user figures out how to do it ("getting started")
- Translating from the language of the problem domain to the language of actions upon user interface objects
- Example areas:
  - **Existence** (of a way to carry out intention or of a cognitive affordance to indicate it
    - Missing feature or lack of physical affordance
    - Missing visual cue or indicator to show the way

9

Translation (cont'd)



- **Meaning / effectiveness** of cognitive affordance to aid Translation (what it says)
  - Clarity, completeness, error avoidance, relevance, consistency, compliance
- **Presentation / Appearance** of cognitive affordance to aid Translation
  - Legibility, noticeability, timing, layout, grouping, complexity, consistency
- **Task structure and interaction control**
  - Alternatives and shortcuts, consistency, efficiency, direct manipulation support

10

About Physical Actions

- All user inputs to operate controls and manipulate objects within the user interface (e.g., clicking, typing, dragging)
- Example areas:
  - Perceiving affordances
    - Noticeability, legibility, contrast, timing
  - Manipulating affordances
    - Interaction complexity, I/O devices, interaction styles and techniques
    - Physical control, manual dexterity, layout (Fitts' law), physical disabilities

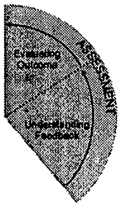
11

About Outcome

- Internal state change within system due to the user action
- User normally infers the outcome based on system response, through feedback
- Example areas:
  - System automation
  - Locus of control
  - System is presumptuous about what the user wanted
  - System errors

12

About Assessment



- Evaluate what happened and the favorability or desirability of the outcome
- How feedback is perceived, understood, and used to assess the outcome of a user action
- Example areas:
  - Existence of feedback
    - Missing, unnecessary, not expected
  - Presentation/Appearance of feedback
    - Legibility, noticeability, timing, complexity
  - Meaning/effectiveness of feedback content
    - Clarity, completeness, relevance, correctness

13

Points About Interaction

- **Can go around cycle at almost any level of task/action granularity (can skip actions not significant to analysis)**
- **Users rarely work out many tasks, intentions in advance; done one at a time**
  - E.g., user had intention to "invoke print command", but did not have an intention to bring up print dialogue box. That just happened in the course of interaction and led to an intention to deal with the dialogue box as part of incremental planning at top.
- **Special attention is given to translating the first intention for each task, the 'getting started' intention**
  - Often the hardest for the user (and the designer) to map to an obvious action. After that, the user can often follow the course of interaction.
- **Making translation to action usually involves a decision, requires user knowledge**

14

Flow of Interaction Cycle

- **Typical flow is sequential around cycle**
- **But parts can appear out of order, some skipped, some repeated**
- **Expert users may "automate" planning, going directly to physical actions**
- **Flow can be user initiated or initiated by environment, system, or other users**
- **Secondary tasks arise for error recovery, exploration, learning, and finding information or other supplemental resources needed to complete primary tasks**

15

Using the UPI to Discover Problems

- **Noting a potential problem for a particular task is based on answering:**
  - A series of questions in the UPI that traverses areas in the Interaction Cycle
- **By selecting "Yes" on a particular question, inspector is taken deeper into the taxonomy of usability attributes**
- **At an end-node, inspector fills out a Usability Problem Report**
- **By selecting "No", inspector continues to traverse each area of the Interaction Cycle until:**
  - A problem is noted
  - Inspector reaches end of UPI questions and concludes that no problems exist for that particular task

16

Example: Formatting a Document

- **User working on business letter goal, new task arises: format document on page**
- **First intention, the "getting started intention": user intends to set the left margin**
- **Translation: user might think to click on the ruler to place current margin symbol**
  - User discovers that the margin symbol is not manipulable
  - There is no affordance to support this mapping to an **action**
  - Example path for usability problem:
    - Planning / Translation / Existence of a way/ Lack of physical affordance

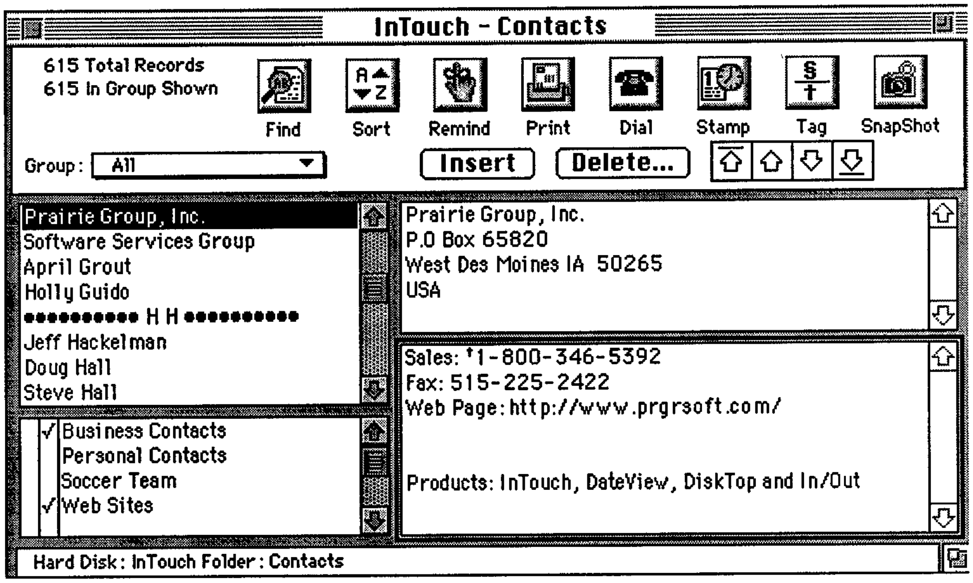
Errors in Tasks

- **Alternative task scenario:**
  - User **evaluates** outcome: Feedback does not makes sense at this point in the interaction; task is off track
  - Formulate next **task**: error recovery
  - Formulate next **intention**: User intends to take first step in error recovery (**secondary task**)
  - Translation: User must now figure out what to do first to recover from the error situation
    - Needs cognitive affordance, same as for primary task intentions
- **And so on, around the cycle**

Key Terms

- **Cognitive affordance (visual cues to see a button)**
  - Aids for knowing and understanding
  - Aids to show the way
- **Physical affordance (a button that can be "clicked")**
  - Aids for doing
- **Example**
  - A chair provides both. Physical affordance of a chair allows sitting on it. Cognitive affordance of a chair lets user see that it is something to sit on
- **Effective affordances support the users' ability to plan physical actions to carry out intentions**

InTouch Information Sheet for UPI Participants



615 Total Records
615 In Group Shown

Prairie Group, Inc.
Software Services Group
April Grout
Holly Guido
●●●●●●●●●● H H ●●●●●●●●●●
Jeff Hackelman
Doug Hall
Steve Hall

✓ Business Contacts
Personal Contacts
Soccer Team
✓ Web Sites

Prairie Group, Inc.
P.O Box 65820
West Des Moines IA 50265
USA

Sales: †1-800-346-5392
Fax: 515-225-2422
Web Page: http://www.prgrsoft.com/

Products: InTouch, DateView, DiskTop and In/Out

Hard Disk: InTouch Folder: Contacts

InTouch is an Address Book...

InTouch is a free-form information manager. Instead of separate fields for each piece of information, InTouch uses just two data entry fields. Most people use InTouch as a contact manager and use the first field for names and addresses and the second field for phone numbers and notes. The second field can hold up to 14 pages of notes that can be time and date stamped to allow you to track conversations, projects or other notations.

InTouch is Free-form...

InTouch offers a refreshing alternative to predefined structured personal information managers or "design your own" databases for keeping information organized with your Macintosh. Sophisticated, but simple to use, InTouch offers a very straight forward way to put information to use.

Free-form is better...

The big advantage to free-form is that you can enter data very rapidly, and InTouch gladly takes what ever information you type. You don't have to worry about how long an address is or how many lines it takes. Foreign addresses present no problem for InTouch. You simply use as many lines as you need, in the order that makes sense to you.

Instructions for UPI Participants

1. You will be using the Usability Problem Inspector (UPI) to evaluate the InTouch interface.
2. Please use the web page forms while inspecting the InTouch interface.
3. If the task is not clear or you are not sure how to perform the task using InTouch, please ask the researcher.
4. The task-based inspection should take approximately one hour to complete.
5. After completing the task-based inspection, you will be given 15 minutes to inspect the InTouch interface using a free-exploration approach. That is, go back and review the menus, dialog boxes, messages associated with the tasks given below.
6. Work as quickly as possible to finish all six tasks and the free exploration within the time allotted.
7. The task list given below is representative of what a typical user may have to perform.

Task 1. Insert a Record

The file that you see is your family rolodex. Put your name, address, and home phone number in the rolodex so other family members can contact you in an emergency.

Task 2. Save a File under a different name

Save a copy of your family rolodex for personal use. Call the copy "MY ROLODEX". You will work with "MY ROLODEX" for the remainder of this session.

Task 3. Find a specific Record

Ken Landry is already entered in your rolodex. He just called you with his new home phone number, add it to his entry. Ken's home phone number is 500-1234.

Task 4. Sort a File

Order the entries in your rolodex alphabetically so that people with the same last name appear together, with family members listed alphabetically by first name.

Task 5. Make a new Group

You notice that all your relatives have "RELATIVE" in the notes field. You decide to make "family" a new group, and make all your relatives members of the group.

Task 6. Import data from a File

You're in charge of organizing soccer this year. Last year's organizer gave you a tab-delimited text file ("Soccer People") containing the names and addresses of the relevant people. Add these people to your rolodex -- put them in the Soccer group for easy reference.

User-Class Definition for UPI Participants

The typical person using the InTouch rolodex program will have the following characteristics:

1. A Mac user: Uses a Mac at work or home
2. Frequently uses email, internet, and MS Office applications on the Mac
3. Has used a different address book program in the past
4. Technically savvy, has no need to look at a manual unless nothing works
5. Has used the Import function to bring in his address list from another program into the current file
6. Knowledge of typical menu commands on the Mac such as File..Open, Save As, Save, Import, Cut, Copy, Paste, Undo
7. Uses menus to accomplish most tasks
8. Typical age range: 25-50

Appendix K. Training Materials for Cognitive Walkthrough Method

Training Package for Cognitive Walkthrough Method

TRAINING PACKAGE

Cognitive Walkthrough
Inspection Study

October 1999

DIRECTIONS: This packet contains a brief training package to reacquaint you to the Cognitive Walkthrough. Even if you consider yourself to be an expert user with this technique, please take a moment to review this package before beginning the inspection process. This training package should take approximately 20-30 minutes to complete. After completing this training package, you will be introduced to the interface application and instructed on how to begin your cognitive walkthrough of the interface.

Cognitive Walkthrough - Background

The Cognitive Walkthrough is a review technique where expert evaluators construct task scenarios from a specification or early prototype and then role play the part of a user working with that interface--"walking through" the interface.

Evaluators scrutinize each step along the way, noting where the interface blocks the "user" from completing the task.

Telling the Story

For Each Action in the Task

- Tell a story that explains how users will choose the correct action...

  ...or that points up why they will go wrong.

Like the solution to a crime in a mystery novel, each story has to provide a credible explanation of what will happen, that includes

- motive
- opportunity
- information that might lead or mislead the actor

Inputs to a Cognitive Walkthrough

Detailed Description of User Interface

Suite of Tasks for Evaluation

For Each Task

- User's description of the task
- Sequence of actions required by the interface to perform the task

Explicit Assumptions About User Population

- User's knowledge about the task
- User's knowledge about the interface

Actions

May be:

- Simple Motor Movements, i.e. Low Level Actions

- Press'#'
- Move cursor with mouse to 'File' Menu

Or:

- Well Integrated Sequences of Low Level Actions

- Select "Save" From "File" Menu
- Enter Password

Level of Description Should Be a Function of:

- User's background
- Action definition assumed by interface

- Level of Description in Prompts

Outline of Theory of Use and Learning by Exploration

Users Start With a Rough Description of What They Want to Accomplish (A Task)

Users Explore Interface to Discover Actions Useful In Accomplishing Their Current Task

Users Select Actions That They Think Will Accomplish Their Current Task

- Often based on match between what they are trying to do and descriptions of actions

Users Assess Progress by Trying to Understand System Responses

- To decide if what they just did was correct action
- To get clues for next correct action

Interface Features

- Link actions to possible user intentions

- e.g. menus signal progress

Suggest new actions

- e.g. prompts

Outline of Cognitive Walkthrough

For a Good Design:

- We can tell a step-by-step story showing how the interface guides the user in
- deciding what to do next, and
- choosing a correct action to do it

Walkthrough Checks Credibility of Story by Examining Each Step

- Will the user try to achieve the right effect?
- Will the user notice that the correct action is available?
- Will the user know that the correct action will achieve the desired effect?
- If the correct action is performed, will the user see that things are going OK?
- If not, they may...

    try new actions for something they have already done, or

    bail out

Examples of Credible Success Stories

Example: An experienced Macintosh user starts an application by double clicking its icon.

- User knows to start the application because they know you have to start an application to use it.
- User knows that double clicking is possible from experience.
- They know that double clicking is the action to use from experience.
- The changes to the display and menu bar signal that the application has started.
- Note that the first three parts of this story would not work for a person new to computers, and the second and third would not work for people without Mac experience.

Examples of Credible Success Stories (Cont.)

Example. An experienced Macintosh user pulls down the GRAPH menu in preparing a graph in a presentation graphics package.

- User is trying to prepare a graph because that is the overall task.
- User knows to pull down this menu because the title GRAPH is clearly related to what they are trying to do.
- User knows that pulling down the menu is possible, and that that is the action to take if the label looks good, by experience with the Mac.
- User knows things are going OK when they see a palette of graph types on the pulldown menu.

Common Features from These (and Other) Success Stories

Users May Know What Effect To Achieve:

- Because it is part of their original task, or
- Because they have experience using a system, or
- Because the system tells them to do it

Users May Know An Action Is Available:

- By experience, or
- By seeing some device (like a button) or
- By seeing a representation of an action (like a menu entry)

Users May Know An Action Is Appropriate To Do What They Are Trying To Achieve:

- By experience, or
- Because the interface provides a prompt or label that connects the action to what they are trying to do, or
- Because all other actions look wrong

Users May Know Things Are Going OK After An Action:

- By experience, or

By recognizing a connection between a system response and what they were trying to do

Success vs. Failure Stories

When You *Can* Tell a Good Story About an Action:

- You may not need to record the story
- But you may want to record any assumptions about what users need to know

Real Payoff From the Walkthrough Is Noting Cases Where You *Can NOT* Tell a Good Story.

- No good story means the interface breaks down
- Need to record where and why the story fails
  - ➢ to identify problem spots to designers
  - ➢ to guide designers in correcting problems

Success Stories require success under all four of the analysts' criteria, while failure stories typically fail under a single criterion.

Failure Stories - Some Examples

Will the User Be Trying to Achieve the Right Effect?

- In an early office system it was necessary to clear a field on a menu by pressing a special key before typing into it. Learners did not know they needed to do this.

Will the User Know That the Correct Action Is Available? In a particular graphing program, changing font and other characteristics of the graph title is achieved by double-clicking on the title to open a dialog box. Users often do not consider double-clicking in this task context.

Failure Story Examples (Cont.)

Will the User Know That the Correct Action Will Achieve the Desired Effect?

- On one of the early popular word-processing systems some key operations, like printing, were accessed from a special menu. But to make this menu appear users had to press a special key, labeled REQ.
- Type styles in one word processor are on a menu called FORMAT. Will users go for the right menu if they are trying to put something in italics? Also consider that there is another menu called FONT.
- An audio prompt may tell the user to "press the pound sign" on a phone keypad. Will the user know that the symbol "#" on the key pad is the pound sign?

If the Correct Action Is Made, Will the User See That Things Are Going OK?

- An early office system had a menu for signing off. When you signed off, the sign-on menu appeared. Some users filled this in, and were caught in a loop.
- In the same system no feedback was presented when a document was printed (remotely) unless the user requested it explicitly. Some learners printed documents repeatedly, not knowing they had succeeded.

Walkthrough Example

Users: The larger class of users includes all staff and faculty on the university's campus, plus their guests and other visitors. It is expected that they have used either a touch-tone or rotary dial telephone. We assume for this evaluation that one of the users is a professor. The professor has used the phone system several times to place outgoing and receive incoming calls. The professor further knows that you can program your phone to do assorted tasks such as forwarding your calls.

Task: I want my phone calls to be forwarded to my associate's office. My associate's number is 492- 1234.

Action sequence: Required Actions and Responses on CU Phone System:

1. Pick up the receiver.

Phone: *dial tone*

2. Press #2 (Command to Cancel Forwarding)

Phone: *bip bip bip*

3. Hang up the receiver

4. Pick up the receiver

Phone: *dial tone*

5. Press *2 (Command to Forward Calls)

Phone: *dial tone*

6. Press 21234

Phone: *bip bip bip*

7. Hang up the receiver.

Interface: There is a Template (Assume it Has Not Been Mislaid) that Includes the Following Material:

FWD *2

CNCL#2

SEND ALL *3

CNCL#2

The Walkthrough: Step-by-Step Analysis Phase

1. Pick up the receiver.

   Phone: *dial tone*

   Success story: This seems OK based on prior experience with phones. But note that there are now phones that you "program" without picking them up!

2. Press #2 (Command to Cancel Forwarding)

   Phone: *bip bip bip*

   Failure story:

   Criterion: Will the User Be Trying to Achieve the Right Effect?

   Big trouble here. Why would user be trying to cancel forwarding? They just have to know.

   Criterion: Will the User Know That the Correct Action Will Achieve the Desired Effect?

   Even if they know to do this they might not recognize CNCL on the template, and they might think the required action is pressing just the "number" 2, not "#" and 2. And they might try to press these buttons together rather than in order.

   Criterion: If the Correct Action is Taken, Will the User See that Things are Going OK?

   Furthermore, how do users know they've succeeded? After experience you recognize *the bips* as a confirmation, but will they at first?

Step-by-Step Analysis Phase (Cont'd)

3. Hang up the receiver

   Failure story:

   Criterion: Will the User Be Trying to Achieve the Right Effect?

   More big trouble. Even if you know you have to cancel forwarding, why should you have to hang up before reestablishing it?

4. Pickup the receiver

   Phone: *dial tone*

   Success story: This seems OK based on experience (but remember that not all phones require this now).

5. Press *2 (Command to Forward Calls)

   Phone: *dial tone*

   Failure story:

   Criterion: Will the User Know That the Correct Action Will Achieve the Desired Effect?

   Here the issue is deciding between *2 and *3. The description on the template is of little help: some people won't recognize FWD and SEND ALL will look good even if they do.

   Criterion: If the Correct Action is Taken, Will the User See that Things are Going OK?

Step-by-Step Analysis Phase (Cont'd)

6. Press 21234

Phone: *bip bip bip*

Failure story:

Criterion: Will the User Be Trying to Achieve the Right Effect?

How does the user know to enter the number now? It's maybe not an unreasonable guess, but the dial tone doesn't constitute much guidance because it only suggests that the phone is active.

Criterion: Will the User Know That the Correct Action Will Achieve the Desired Effect?

Also, there is a likelihood of error in not working out the *form of* the number that is needed. That is the user must understand that it is sufficient and correct to enter "21234" and that the entire number sequence of "4921234" is not needed.

Criterion: If the Correct Action is Taken, Will the User See that Things are Going OK?

As above, the bips may not mean much to someone starting out.

7. Hang up the receiver.

Success story: Seems OK based on prior experience with phones.

Follow the Action Sequence

Track Correct Actions

- Even if there is a big problem in the interface, the analysis tracks the path of the correct actions, NOT what users may really do.

Reset After Problems

- The analysis "resets" after noting a problem and goes on as if the problem had not occurred.

- *always* assume correct action was taken, so display and system are ready for next action
- *may* have to assume a "fix" to make sense of rest of waikthrough - for example:
  - ▪ assume user had missing knowledge, which might be useful in later actions
  - ▪ or, assume change to system so knowledge wasn't needed.

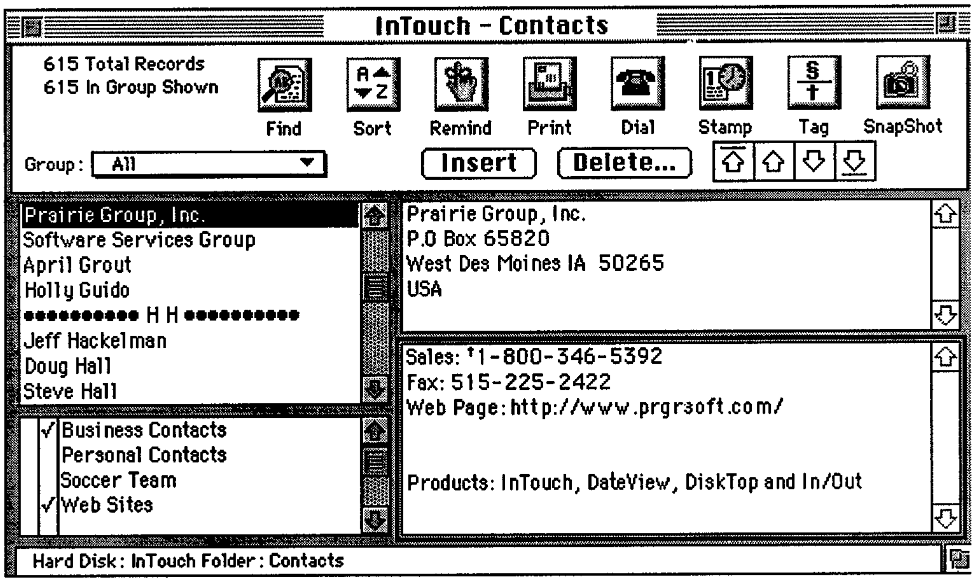
"Label Following" Strategy

Look for Word Matches

- Things work well when system prompts use the same words or labels as user goals.

If words don't match, user knowledge is required.

InTouch Information Sheet for Cognitive Walkthrough Participants



615 Total Records
615 In Group Shown

Group: All

Prairie Group, Inc.
Software Services Group
April Grout
Holly Guido
●●●●●●●●●● H H ●●●●●●●●●●
Jeff Hackelman
Doug Hall
Steve Hall

✓ Business Contacts
Personal Contacts
Soccer Team
✓ Web Sites

Prairie Group, Inc.
P.O Box 65820
West Des Moines IA 50265
USA

Sales: *1-800-346-5392
Fax: 515-225-2422
Web Page: http://www.prgrsoft.com/

Products: InTouch, DateView, DiskTop and In/Out

Hard Disk: InTouch Folder: Contacts

InTouch is an Address Book...

InTouch is a free-form information manager. Instead of separate fields for each piece of information, InTouch uses just two data entry fields. Most people use InTouch as a contact manager and use the first field for names and addresses and the second field for phone numbers and notes. The second field can hold up to 14 pages of notes that can be time and date stamped to allow you to track conversations, projects or other notations.

InTouch is Free-form...

InTouch offers a refreshing alternative to predefined structured personal information managers or "design your own" databases for keeping information organized with your Macintosh. Sophisticated, but simple to use, InTouch offers a very straight forward way to put information to use.

Free-form is better...

The big advantage to free-form is that you can enter data very rapidly, and InTouch gladly takes what ever information you type. You don't have to worry about how long an address is or how many lines it takes. Foreign addresses present no problem for InTouch. You simply use as many lines as you need, in the order that makes sense to you.

Instructions for Cognitive Walkthrough Participants

1. You will be using the Cognitive Walkthrough technique to evaluate the InTouch interface.
2. Please follow the tasks and steps provided in the Cognitive Walkthrough sheets.
3. If the task is not clear or you are not sure how to perform the task using InTouch, please ask the researcher.
4. The cognitive walkthrough inspection should take approximately 1 hour and 15 minutes to complete.
5. Work as quickly as possible to finish all six tasks within the time allotted.
6. Although alternative paths may be possible for a given task, only examine the steps provided in the Cognitive Walkthrough sheets.
7. The task list given below is representative of what a typical user may have to perform (these task descriptions and the associated steps are provided in the Cognitive Walkthrough forms).

Task 1. Insert a Record

The file that you see is your family rolodex. Put your name, address, and home phone number in the rolodex so other family members can contact you in an emergency.

Task 2. Save a File under a different name

Save a copy of your family rolodex for personal use. Call the copy "MY ROLODEX". You will work with "MY ROLODEX" for the remainder of this session.

Task 3. Find a specific Record

Ken Landry is already entered in your rolodex. He just called you with his new home phone number, add it to his entry. Ken's home phone number is 500-1234.

Task 4. Sort a File

Order the entries in your rolodex alphabetically so that people with the same last name appear together, with family members listed alphabetically by first name.

Task 5. Make a new Group

You notice that all your relatives have "RELATIVE" in the notes field. You decide to make "family" a new group, and make all your relatives members of the group.

Task 6. Import data from a File

You're in charge of organizing soccer this year. Last year's organizer gave you a tab-delimited text file ("Soccer People") containing the names and addresses of the relevant people. Add these people to your rolodex -- put them in the Soccer group for easy reference.

User-Class Definition for Cognitive Walkthrough Participants

The typical person using the InTouch rolodex program will have the following characteristics:

1. A Mac user: Uses a Mac at work or home
2. Frequently uses email, internet, and MS Office applications on the Mac
3. Has used a different address book program in the past
4. Technically savvy, has no need to look at a manual unless nothing works
5. Has used the Import function to bring in his address list from another program into the current file
6. Knowledge of typical menu commands on the Mac such as File..Open, Save As, Save, Import, Cut, Copy, Paste, Undo
7. Uses menus to accomplish most tasks
8. Typical age range: 25-50

Appendix L. Training Materials for Heuristic Evaluation Method

TRAINING PACKAGE

Heuristic
Inspection Study

October 1999

DIRECTIONS: This packet contains a brief training package to reacquaint you to Heuristic Evaluation for user interfaces. Even if you consider yourself to be an expert user with this technique, please take a moment to review this package before beginning the inspection process. This training package should take approximately 20-30 minutes to complete. After completing this training package, you will be introduced to the interface application and instructed on how to begin your heuristic evaluation of the interface.

What is Heuristic Evaluation?

Heuristic evaluation (Nielsen and Molich, 1990; Nielsen 1994) is a usability engineering method for finding the usability problems in a user interface design so that they can be attended to as part of an iterative design process. Heuristic evaluation involves having a small set of evaluators examine the interface and judge its compliance with recognized usability principles (the "heuristics").

A heuristic is a guideline or general principle or rule of thumb that can guide a design decision or be used to critique a decision that has already been made. Heuristic evaluations help to structure the critique of a system using a set of relatively simple and general heuristics.

The general idea behind heuristic evaluation is that several evaluators independently evaluate a system to come up with potential usability problems. It is important that there be several of these evaluators and that the evaluations be done independently. Nielsen's experience indicates that around 5 evaluators usually results in about 75% of the overall usability problems being discovered.

What is evaluated? Heuristic evaluation is best used as a design time evaluation technique, because it is easier to fix a lot of the usability problems that arise. But all that is really required to do the evaluation is some sort of artifact that describes the system, and that can range from a set of storyboards giving a quick overview of the system all the way to a fully functioning system that is in use in the field.

During the evaluation session, the evaluator goes through the interface several times and inspects the various dialogue elements and compares them with a list of recognized usability principles (the heuristics). These heuristics are general rules that seem to describe common properties of usable interfaces. In addition to the checklist of general heuristics to be considered for all dialogue elements, the evaluator obviously is also allowed to consider any additional usability principles or results that come to mind that may be relevant for any specific dialogue element. In many cases, the evaluators are given typical usage scenarios to help understand the primary tasks that users accomplish with the system.

The output from using the heuristic evaluation method is a list of usability problems in the interface with references to those usability principles that were violated by the design in each case in the opinion of the evaluator. It is not sufficient for evaluators to simply say that they do not like something; they should explain why they do not like it with reference to the heuristics or to other usability results. The evaluators should try to be as specific as possible and should list each usability problem separately. For example, if there are three things wrong with a certain dialogue element, all three should be listed with reference to the various usability principles that explain why each particular aspect of the interface element is a usability problem. There are two main reasons to note each problem separately: First, there is a risk of repeating some problematic aspect of a dialogue element, even if it were to be completely replaced with a new design, unless one is aware of all its problems. Second, it may not be possible to fix all usability problems in an interface element or to replace it with a new design, but it could still be possible to fix some of the problems if they are all known.

Heuristic evaluation does not provide a systematic way to generate fixes to the usability problems or a way to assess the probable quality of any redesigns. However, because heuristic evaluation aims at explaining each observed usability problem with reference to established usability principles, it will often be fairly easy to generate a revised design according to the guidelines provided by the violated principle for good interactive systems. Also, many usability problems have fairly obvious fixes as soon as they have been identified.

Nielsen's Heuristics

- ❑ **Visibility of system status**

  The system should always keep the user's informed about what is going on, through appropriate feedback within reasonable time

- ❑ **Match between system and the real world**

  The system should speak the users' language, with words, phrases, and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order

- ❑ **User control and freedom**

  Users often choose system functions by mistake and will need a clearly marked 'emergency exit' to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

- ❑ **Consistency and standards**

  Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

- ❑ **Error prevention**

  Even better than good error messages is a careful design which prevents a problem from occurring in the first place.

- ❑ **Recognition rather than recall**

  Make objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

- ❑ **Flexibility and efficiency of use**

  Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

- ❑ **Aesthetic and minimalist design**

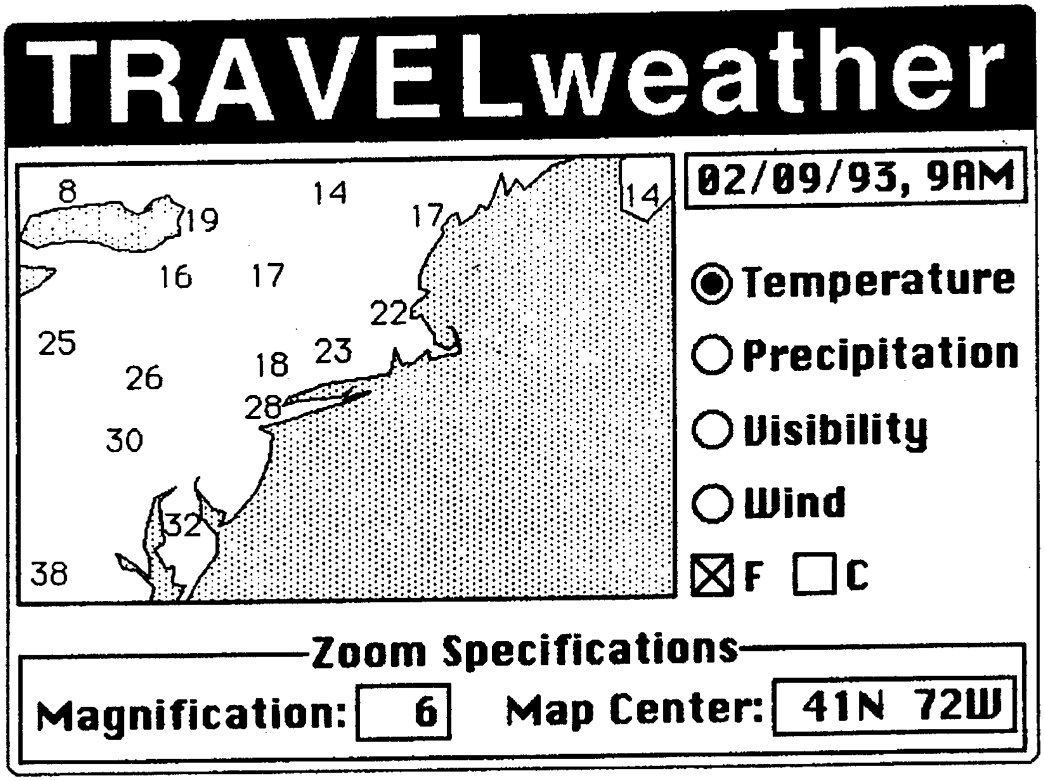
  Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

- ❑ **Help users recognize, diagnose, and recover from errors**

  Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

In order to illustrate each of Nielsen's Heuristics, we employ the following heuristic evaluation of a paper mock-up:

Shown below in Figure 1 is the design for a system to provide weather information to travelers. TRAVELweather (a non-existing system) can provide information about the weather at 3AM, 9AM, 3PM, and 9PM for the current day as well as the two next days, using reported readings for past weather and forecasts to predict future weather. The interface is designed for use on a graphical personal computer with a mouse, and will appear in a separate window on the screen.



TRAVELweather

02/09/93, 9AM

● Temperature

○ Precipitation

○ Visibility

○ Wind

☒ F □ C

Zoom Specifications

Magnification: 6

Map Center: 41N 72W

Figure 1. Screen design for a hypothetical system to provide weather information and forecasts to travelers.

The user operates the interface by typing the desired time into the box in the upper right part of the screen. If the user types a date other than today or the next two days, or if the user types a time other than the four times for which information is available, the system will show an alert dialog box with the following error message: "Weather Data Not Available." The only button in the error message box is an "OK" button. Clicking the OK button will make the dialog box go away and will reset the date and time specification to the previous value.

The user changes the map display by editing the boxes for zoom magnification and for the center of the map. The system ensures that only integer numbers can be typed in the map magnification box by simply beeping every time the user presses a non-number key in that box. If the user types anything in the map center box other than a valid set of coordinates (an integer from 0 to 90 followed by the letter N or S followed by an integer from 0 to 179 followed by the letter W or E), the system will show an alert dialog box with the following error message: "Unknown Map Coordinates." The only button in the error message box is an "OK" button. Clicking the OK button will make the dialog box go away and will reset the coordinates to their previous value.

With respect to all three input boxes, the user's changes take effect as soon as the user clicks the mouse outside a box after having edited it.

Based upon this mock-up, we will now show an example of a usability problem that corresponds to each of Nielsen's Heuristics. It is worth noting that not all of the system's shortcomings are discussed. For the purpose of this illustration, only one example was chosen for each heuristic.

Visibility of system status

The system should always keep the user's informed about what is going on, through appropriate feedback within reasonable time

The map should display the names of at least some larger cities and other locations of interest to allow users to better recognize these locations. One way of including additional names without cluttering up the map would be to pop up the names of cities close to the weather stations when the user slides the mouse over a weather reading.

Match between system and the real world

The system should speak the users' language, with words, phrases, and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order

The pattern used to denote oceans and lakes does not make it sufficiently clear what parts of the map are land and what are water. Instead of the current pattern, use a wavy pattern (or blue on a color screen).

User control and freedom

Users often choose system functions by mistake and will need a clearly marked 'emergency exit' to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

Users should not be punished for making errors by having the system delete all their input. Instead, the erroneous user input should be retained to allow the user to edit it. Alternatively, to keep the fields on the main screen correct, repeat the erroneous input in the error dialog box and allow users to edit it there.

Consistency and standards

Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

Error prevention

Even better than good error messages is a careful design which prevents a problem from occurring in the first place.

Requiring the user to click outside the entry box before changes will take effect is error prone. It is likely that many users will forget this and will wonder why nothing happens after they changed the text. One possible way to reduce the likelihood of this error is to have an explicit "do it" button. Also, the user's changes should lake effect if the user hits the enter or return keys. Redesigning the interface as suggested above to replace the text entry boxes with a combination of pop-up menus, scroll bars, zoom buttons, and a click shortcut would also solve the problem.

Recognition rather than recall

Make objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

No options are presented for displaying the appropriate instructions for operating the system.

Flexibility and efficiency of use

Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

It is not clear from the specification to what extent the system will be used repeatedly by the same users (home or office use) or whether it will be used mainly by a flow of changing users (airport etc. use). If the same users can be expected to repeatedly use the system, they will probably also repeatedly ask for weather for the same areas. Support for this user need can be provided by having the system remember the last seven or so locations typed in the map center box and provide direct access to them through a pop-up menu. The next time the system was started, the map could also come up with the zoom specifications (magnification and center) set to the values from the last time the same user used the system.

Aesthetic and minimalist design

Dialogues should not contain information that is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

The name of the system is displayed much too prominently. By making the name smaller, room could be provided for alternative dialogue elements, or the screen could be made less busy.

Help users recognize, diagnose, and recover from errors

Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

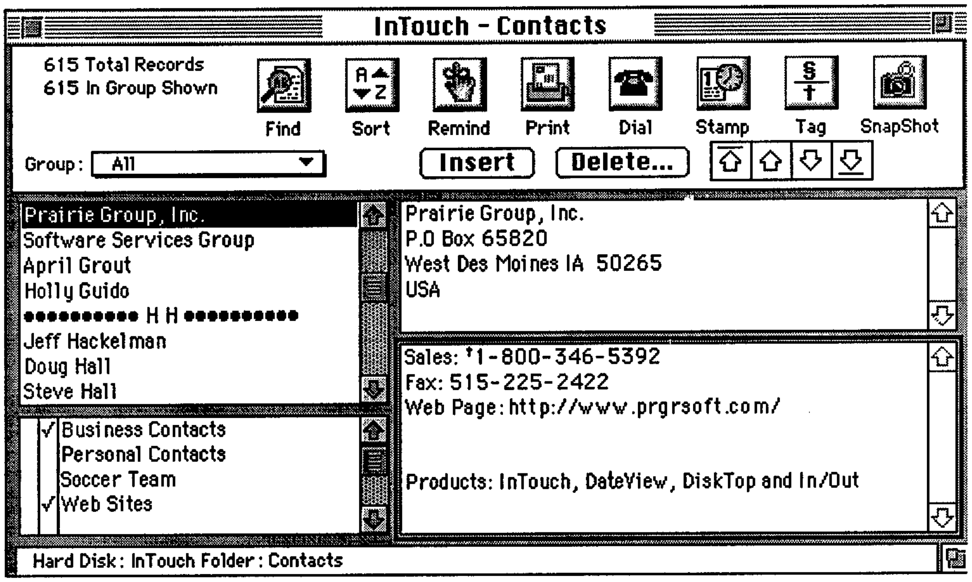
The error message "Weather data not available" is not precise. Instead, the system should repeat the date and time as entered by the user and explain why they were not acceptable to the system. Different error messages should be used for dates and times that are not formatted correctly, dates and times that are before or after the time interval for which weather information is available, and times that are not one of the four hours for which information is available.

Help and documentation

Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

No user help or assistance is provided for this system.

InTouch Information Sheet for Cognitive Walkthrough Participants



615 Total Records
615 In Group Shown

Group: All

Prairie Group, Inc.
Software Services Group
April Grout
Holly Guido
●●●●●●●●●● H H ●●●●●●●●●●
Jeff Hackelman
Doug Hall
Steve Hall

✓ Business Contacts
Personal Contacts
Soccer Team
✓ Web Sites

Prairie Group, Inc.
P.O Box 65820
West Des Moines IA 50265
USA

Sales: *1-800-346-5392
Fax: 515-225-2422
Web Page: http://www.prgrsoft.com/

Products: InTouch, DateView, DiskTop and In/Out

Hard Disk: InTouch Folder: Contacts

InTouch is an Address Book...

InTouch is a free-form information manager. Instead of separate fields for each piece of information, InTouch uses just two data entry fields. Most people use InTouch as a contact manager and use the first field for names and addresses and the second field for phone numbers and notes. The second field can hold up to 14 pages of notes that can be time and date stamped to allow you to track conversations, projects or other notations.

InTouch is Free-form...

InTouch offers a refreshing alternative to predefined structured personal information managers or "design your own" databases for keeping information organized with your Macintosh. Sophisticated, but simple to use, InTouch offers a very straight forward way to put information to use.

Free-form is better...

The big advantage to free-form is that you can enter data very rapidly, and InTouch gladly takes what ever information you type. You don't have to worry about how long an address is or how many lines it takes. Foreign addresses present no problem for InTouch. You simply use as many lines as you need, in the order that makes sense to you.

Instructions for Heuristic Evaluation Participants

1. You will be using the Heuristic Evaluation technique to evaluate the InTouch interface.
2. Please use the Heuristic Evaluation forms while inspecting the InTouch interface.
3. If the task is not clear or you are not sure how to perform the task using InTouch, please ask the researcher.
4. The inspection should take approximately one hour and 15 minutes to complete.
5. Work as quickly as possible to finish your evaluation within the time allotted.
6. The usage scenario described below is representative of the typical tasks a user may have to perform.
7. Users of InTouch will frequently perform the tasks described below. You should focus your heuristic evaluation on the types of commands, menus, dialog boxes, and icons that a user will most likely use to accomplish these tasks.

Inserting a Record

Users will want to insert name, address, and phone numbers for new entries.

Saving a File under a different name

Users will save a file under a new name, maybe using different file names for specific purposes.

Finding a specific Record

As the address book records increase in number, users will need to use the Find command to look for a particular name or note.

Sorting a File

In addition to using Find, users may find the Sort command helpful to see a list of people with the same last name.

Making a new Group

Users may want to group certain records together based on a specific word in the notes field. Thus, they will want to add new groups and move specific records to the specified group.

Importing data from a File

Users may be sent tab-delimited files from friends with a list of names and addresses. Users will want to be able to import these addresses rather than manually entering every record.

User-Class Definition for Heuristic Evaluation Participants

The typical person using the InTouch rolodex program will have the following characteristics:

1. A Mac user: Uses a Mac at work or home
2. Frequently uses email, internet, and MS Office applications on the Mac
3. Has used a different address book program in the past
4. Technically savvy, has no need to look at a manual unless nothing works
5. Has used the Import function to bring in his address list from another program into the current file
6. Knowledge of typical menu commands on the Mac such as File..Open, Save As, Save, Import, Cut, Copy, Paste, Undo
7. Uses menus to accomplish most tasks
8. Typical age range: 25-50

Appendix M. Comparison Study Post-Test Questionnaire

Comparison Study Post-Test Questionnaire

Directions:

Please indicate how strongly you disagree or agree to the statements using the scale provided.

	strongly disagree	disagree	neutral	agree	strongly agree	NA
1. The ________________ technique helped me to effectively evaluate the InTouch interface.	○	○	○	○	○	○
2. The ________________ technique helped me to quickly evaluate the InTouch interface.	○	○	○	○	○	○
3. The ________________ technique was easy to learn.	○	○	○	○	○	○
4. The ________________ technique was easy to apply.	○	○	○	○	○	○
5. I would recommend using the ________________ technique for performing usability inspection in my organization.	○	○	○	○	○	○

Comments on evaluation technique:

____

____

____

____

____

____

____

____

____

____

____

____

____

VITA

TERENCE S. ANDRE

EDUCATION

M.S. 1991
California Polytechnic State University, San Luis Obispo, California
Major: Industrial Engineering/Human Factors

B. S. 1987
United States Air Force Academy, Colorado Springs, Colorado
Major: Behavioral Sciences/Human Factors

ACADEMIC AND PROFESSIONAL EXPERIENCE

1997 - Present
Enrolled as a graduate student at Virginia Polytechnic Institute and State University, Blacksburg, Virginia

PhD Candidate, Industrial and System Engineering
Concentration: Human Factors Engineering

1996 - 1997
Assistant Professor
Department of Behavioral Sciences and Leadership
United States Air Force Academy
Colorado Springs, Colorado

1994 - 1996
Instructor
Department of Behavioral Sciences and Leadership
United States Air Force Academy
Colorado Springs, Colorado

1992 - 1994
Human Factors Scientist
Headquarters Air Force Operational Test and Evaluation Center
Kirtland Air Force Base, New Mexico

1991 - 1992
Adjunct Instructor
Department of Psychology
Chapman University
Lompoc, California

1990 - 1992
Human Factors Evaluator
Missile Test Team
Air Force Operational Test and Evaluation Center
Vandenberg Air Force Base, California

ACADEMIC AND PROFESSIONAL EXPERIENCE (continued)

1988 - 1990
Business Manager
Space Launch Program Office
Vandenberg Air Force Base, California

1987 - 1988
Student Pilot/Human Factors Engineer
Williams Air Force Base, Arizona

AWARDS

1997	Company Grade Officer of the Year, United States Air Force Academy.
1996	Company Grade Officer of the Quarter, United States Air Force Academy.
1995	Company Grade Officer of the Quarter, United States Air Force Academy.
1994	Air Force Meritorious Service Medal.
1994	Level III Certification in Program Management and Test & Evaluation.
1993	Test Team of the Quarter Award, Kirtland Air Force Base.
1991	Air Force Commendation Medal.
1990	Air Force Achievement Award.
1989	Company Grade Officer of the Quarter, Vandenberg Air Force Base.
1989	Acquisition Milestone Achiever Award, Vandenberg Air Force Base.

ACTIVITIES AND PROFESSIONAL SOCIETIES

- Phi Kappa Phi National Honor Society
- Alpha Pi Mu National Honor Society for Industrial Engineering
- Education & Training Committee, Human Factors and Ergonomics Society.
- Air Force Representative, DoD Human Factors Engineering Technical Advisory Group.
- Human Factors and Ergonomics Society.
- Association of Graduates, United States Air Force Academy.

PUBLICATIONS

Hartson, H. R., Andre, T. S., & Williges, R. C. (2000). Evaluating usability evaluation methods. Manuscript submitted for publication.

Andre, T. S., Hartson, H. R., Belz, S. M., & McCreary, F. A. (2000). The User Action Framework: A reliable foundation for usability engineering support tools. Manuscript submitted for publication.

Hartson, H. R., Andre, T. S., Williges, R. W., & Van Rens, L. (1999). The user action framework: A theory-based foundation for inspection and classification of usability problems. In H. Bullinger & J. Ziegler (Eds.), Human-computer interaction: Ergonomics and user interfaces (Vol. 1, pp. 1058-1062). Mahway, NJ: Lawrence Erlbaum.

PUBLICATIONS (continued)

Andre, T. S., & Eisenhut, S. A. (1997). Modeling and simulation in the design process. In T. S. Andre & A. W. Schopper (Eds.), Human factors engineering in system design (pp. 57-77). Dayton, OH: Crew System Ergonomics Information Analysis Center.

Andre, T. S., & Schopper, A. W. (Eds.). (1997). Human factors engineering in system design. Dayton, OH: Crew System Ergonomics Information Analysis Center.

Andre, T. S., & Schreiber, H. G. (1997). Designing for reliability and maintainability. In T. S. Andre & A. W. Schopper (Eds.), Human factors engineering in system design (pp. 199-211). Dayton, OH: Crew System Ergonomics Information Analysis Center.

Meister, D., Andre, T. S., & Aretz, A. J. (1997). System Analysis. In T. S. Andre & A. W. Schopper (Eds.), Human factors engineering in system design (pp. 21-55). Dayton, OH: Crew System Ergonomics Information Analysis Center.

CONFERENCE PROCEEDINGS

Andre, T. S., Belz, S. M., McCreary, F. A., & Hartson, H. R. (in press). Testing a framework for reliable classification of usability problems. In Proceedings of the Human Factors and Ergonomics Society 44th Annual Meeting. Santa Monica, CA: Human Factors and Ergonomics Society.

Andre, T. S., Hartson, H. R., & Williges, R. C. (1999). Expert-based usability inspections: Developing a foundational framework and method. In Proceedings of the 2nd Annual Student's Symposium on Human Factors and Ergonomics of Complex Systems (pp. 142-148). Greensboro, NC: North Carolina A&T State University.

Andre, T. S., Williges, R. C., & Hartson, H. R. (1999). The effectiveness of usability evaluation methods: Determining the appropriate criteria. In Proceedings of the Human Factors and Ergonomics Society 43rd Annual Meeting (pp. 1090-1094). Santa Monica, CA: Human Factors and Ergonomics Society.

Andre, T. S., Kleiner, B. M., & Williges, R. C. (1998). A conceptual model for understanding computer-augmented distributed team communication and decision making. In RTO Human Factors and Medicine Panel (HFM) Symposium on Collaborative Crew Performance in Complex Operational Systems (pp. 23-1 - 23-15). Neuilly, France: NATO Research and Technology Organization.

Andre, T. S. (1997). Using CASHE (Computer-Aided Systems Human Engineering) in the classroom. Poster presented at the Human Factors and Ergonomics Society 41st Annual Meeting, Albuquerque, NM.

CONFERENCE PROCEEDINGS (continued)

Andre, T. S., & Pouraghabagher, A. R. (1995). Evaluation of computer-based progress indicators in the missile launch control center. In Proceedings of the Human Factors and Ergonomics Society 39th Annual Meeting (pp. 40-44). Santa Monica, CA: Human Factors and Ergonomics Society.

Andre T. S., & Charlton, S. G. (1994). Strategy-to-task: Human factors operational test and evaluation at the task-level. In Proceedings of the Human Factors and Ergonomics Society 38th Annual Meeting (pp. 1085-1089). Santa Monica, CA: Human Factors and Ergonomics Society.

Andre, T. S. (1993). Strategy-to-task: How does human factors fit within task-level testing? Paper presented at the Department of Defense Human Factors Engineering Technical Advisory Group Thirtieth Meeting, Dayton, OH.

Andre, T. S. (1991). Integrating computer-based technical manuals in the missile launch control center: An operational study. Poster presented at the Human Factors and Ergonomics Society 35th Annual Meeting, San Francisco, CA.

COURSES TAUGHT

1997	Behavioral Sciences 473 (Human Factors Engineering in System Design) USAF Academy
1996	Behavioral Sciences 373 (Introduction to Human Factors Engineering) USAF Academy
1994 - 1995	Behavioral Sciences 110 (Introductory Psychology) USAF Academy
1991	Psychology 310 (Psychology of Learning and Behavior) Chapman University